



SUSE OpenStack Cloud Crowbar 9

Deployment Guide using Crowbar

Deployment Guide using Crowbar

SUSE OpenStack Cloud Crowbar 9

Publication Date: May 02, 2024

<https://documentation.suse.com> 

Copyright © 2006– 2024 SUSE LLC and contributors. All rights reserved.

Except where otherwise noted, this document is licensed under **Creative Commons Attribution 3.0 License** :

<https://creativecommons.org/licenses/by/3.0/legalcode> .

For SUSE trademarks, see <https://www.suse.com/company/legal/>. All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors nor the translators shall be held liable for possible errors or the consequences thereof.

Contents

About This Guide **xii**

- 1 Available Documentation **xiii**
- 2 Feedback **xiv**
- 3 Documentation Conventions **xv**
- 4 About the Making of This Manual **xvi**

I ARCHITECTURE AND REQUIREMENTS **1**

1 The SUSE OpenStack Cloud Architecture **2**

- 1.1 The Administration Server **3**
- 1.2 The Control Node(s) **4**
- 1.3 The Compute Nodes **5**
- 1.4 The Storage Nodes **6**
- 1.5 The Monitoring Node **6**
- 1.6 HA Setup **7**

2 Considerations and Requirements **9**

- 2.1 Network **9**
 - Network Address Allocation **13** • Network Modes **17** • Accessing the Administration Server via a Bastion Network **21** • DNS and Host Names **21**
- 2.2 Persistent Storage **22**
 - Cloud Storage Services **22** • Storage Hardware Requirements **23**
- 2.3 SSL Encryption **26**

- 2.4 Hardware Requirements 27
 - Administration Server 28 • Control Node 28 • Compute Node 29 • Storage Node 29 • monasca Node 29
- 2.5 Software Requirements 30
 - Optional Component: SUSE Enterprise Storage 31 • Product and Update Repositories 32
- 2.6 High Availability 35
 - High Availability of the Administration Server 35 • High Availability of the Control Node(s) 36 • High Availability of the Compute Node(s) 37 • High Availability of the Storage Node(s) 38 • Cluster Requirements and Recommendations 39 • For More Information 41
- 2.7 Summary: Considerations and Requirements 42
- 2.8 Overview of the SUSE OpenStack Cloud Installation 44

II SETTING UP THE ADMINISTRATION SERVER 45

3 Installing the Administration Server 46

- 3.1 Starting the Operating System Installation 46
- 3.2 Registration and Online Updates 46
- 3.3 Installing the SUSE OpenStack Cloud Crowbar Extension 47
- 3.4 Partitioning 47
- 3.5 Installation Settings 47
 - Software Selection 48 • Firewall Settings 48

4 Installing and Setting Up an SMT Server on the Administration Server (Optional) 49

- 4.1 SMT Installation 49
- 4.2 SMT Configuration 49
- 4.3 Setting up Repository Mirroring on the SMT Server 51
 - Adding Mandatory Repositories 51 • Adding Optional Repositories 52 • Updating the Repositories 52

4.4 For More Information 52

5 Software Repository Setup 53

5.1 Copying the Product Media Repositories 53

5.2 Update and Pool Repositories 57

Repositories Hosted on an SMT Server Installed on the Administration Server 57 • Repositories Hosted on a Remote SMT Server 57 • Repositories Hosted on a SUSE Manager Server 58 • Alternative Ways to Make the Repositories Available 59

5.3 Software Repository Sources for the Administration Server Operating System 59

5.4 Repository Locations 60

6 Service Configuration: Administration Server Network Configuration 64

7 Crowbar Setup 66

7.1 *User Settings* 66

7.2 *Networks* 67

Separating the Admin and the BMC Network 68

7.3 *Network Mode* 69

Setting Up a Bastion Network 71

7.4 *Repositories* 73

7.5 Custom Network Configuration 74

Editing `network.json` 75 • Global Attributes 76 • Interface Map 77 • Interface Map Example 79 • Network

Conduits 80 • Network Conduit Examples 85 • Network Definitions 89 • Providing Access to External Networks 92 • Split Public and Floating Networks on Different VLANs 93 • Adjusting the Maximum Transmission Unit for the Admin and Storage Network 93 • Matching Logical and Physical Interface Names with `network-json-resolve` 95

8	Starting the SUSE OpenStack Cloud Crowbar installation	97
9	Customizing Crowbar	101
9.1	Skip Unready Nodes	101
9.2	Skip Unchanged Nodes	102
9.3	Controlling Chef Restarts Manually	103
9.4	Prevent Automatic Restart	105
III	SETTING UP OPENSTACK NODES AND SERVICES	107
10	The Crowbar Web Interface	108
10.1	Logging In	108
10.2	Overview: Main Elements	109
	Nodes	110
	Barclamps	112
	Utilities	112
	Help	113
10.3	Deploying Barclamp Proposals	114
	Creating, Editing and Deploying Barclamp Proposals	115
	Barclamp Deployment Failure	116
	Deleting a Proposal That Already Has Been Deployed	117
	Queuing/Dequeuing Proposals	118
11	Installing the OpenStack Nodes	119
11.1	Preparations	119
11.2	Node Installation	120
11.3	Converting Existing SUSE Linux Enterprise Server 12 SP4 Machines Into SUSE OpenStack Cloud Nodes	126
11.4	Post-Installation Configuration	127
	Deploying Node Updates with the Updater Barclamp	127
	Configuring Node Updates with the <i>SUSE Manager Client</i> Barclamp	131
	Mounting NFS Shares on a Node	133
	Using an Externally Managed Ceph Cluster	135
	Accessing the Nodes	138
	Enabling SSL	139
11.5	Editing Allocated Nodes	140

12 Deploying the OpenStack Services 143

- 12.1 Deploying designate 144
 - Using PowerDNS Backend 146
- 12.2 Deploying Pacemaker (Optional, HA Setup Only) 155
- 12.3 Deploying the Database 162
 - Deploying MariaDB 163
- 12.4 Deploying RabbitMQ 166
 - HA Setup for RabbitMQ 168 • SSL Configuration for RabbitMQ 168 • Configuring Clients to Send Notifications 170
- 12.5 Deploying keystone 171
 - Authenticating with LDAP 173 • HA Setup for keystone 174 • OpenID Connect Setup for keystone 174
- 12.6 Deploying monasca (Optional) 177
- 12.7 Deploying swift (optional) 187
 - HA Setup for swift 191
- 12.8 Deploying glance 192
 - HA Setup for glance 195
- 12.9 Deploying cinder 195
 - HA Setup for cinder 202
- 12.10 Deploying neutron 203
 - Using Infoblox IPAM Plug-in 207 • HA Setup for neutron 208 • Setting Up Multiple External Networks 208
- 12.11 Deploying nova 211
 - HA Setup for nova 216
- 12.12 Deploying horizon (OpenStack Dashboard) 217
 - HA Setup for horizon 218
- 12.13 Deploying heat (Optional) 219
 - Enabling Identity Trusts Authorization (Optional) 220 • HA Setup for heat 222

- 12.14 Deploying ceilometer (Optional) 222
 - HA Setup for ceilometer 224
- 12.15 Deploying manila 224
 - HA Setup for manila 228
- 12.16 Deploying Tempest (Optional) 228
 - HA Setup for Tempest 231
- 12.17 Deploying Magnum (Optional) 231
 - HA Setup for Magnum 233
- 12.18 Deploying barbican (Optional) 233
 - HA Setup for barbican 235
- 12.19 Deploying sahara 235
 - HA Setup for sahara 236
- 12.20 Deploying Octavia 236
 - Prerequisites 239 • Barclmap raw mode 242 • Migrating Users to Octavia 242 • Migrating Neutron LBaaS Instances to Octavia 243
- 12.21 Deploying ironic (optional) 247
 - Custom View Options 247 • ironic Drivers 248 • Example ironic Network Configuration 248
- 12.22 How to Proceed 260
- 12.23 SUSE Enterprise Storage integration 260
- 12.24 Roles and Services in SUSE OpenStack Cloud Crowbar 264
- 12.25 Crowbar Batch Command 267
 - YAML file format 268 • Top-level proposal attributes 268 • Node Alias Substitutions 269 • Options 270
- 13 Limiting Users' Access Rights 271**
 - 13.1 Editing `policy.json` 271
 - 13.2 Editing `keystone_policy.json` 273
 - 13.3 Adjusting the *keystone* Barclamp Proposal 274

13.4	Adjusting the <i>horizon</i> Barclamp Proposal	274
13.5	Pre-Installed Service Admin Role Components	275
14	Configuration Files for OpenStack Services	277
14.1	Default Configuration Files	277
14.2	Custom Configuration Files	277
14.3	Naming Conventions for Custom Configuration Files	278
14.4	Processing Order of Configuration Files	278
14.5	Restarting with New or Changed Configuration Files	278
14.6	For More Information	279
15	Installing SUSE CaaS Platform heat Templates	280
15.1	SUSE CaaS Platform heat Installation Procedure	280
15.2	Installing SUSE CaaS Platform with Multiple Masters	281
15.3	Enabling the Cloud Provider Integration (CPI) Feature	286
15.4	Register SUSE CaaS Platform Cluster for Software Updates	289
15.5	More Information about SUSE CaaS Platform	290
16	Installing SUSE CaaS Platform v4 using terraform	291
16.1	CaaS v4 deployment on SOC using terraform.	291
IV	SETTING UP NON-OPENSTACK SERVICES	292
17	Deploying the Non-OpenStack Components	293
17.1	Tuning the Crowbar Service	293
17.2	Configuring the NTP Service	294
17.3	Installing and using Salt	294

V	TROUBLESHOOTING AND SUPPORT	296
18	Troubleshooting and Support	297
18.1	FAQ	297
18.2	Support	306
	Applying PTFs (Program Temporary Fixes) Provided by the SUSE L3 Support	307
A	Using Cisco Nexus Switches with neutron	309
A.1	Requirements	309
A.2	Deploying neutron with the Cisco Plugin	310
B	Documentation Updates	314
B.1	April 2018 (Initial Release SUSE OpenStack Cloud Crowbar 8)	314
	Glossary of Terminology and Product Names	315

About This Guide

SUSE® OpenStack Cloud Crowbar is an open source software solution that provides the fundamental capabilities to deploy and manage a cloud infrastructure based on SUSE Linux Enterprise. SUSE OpenStack Cloud Crowbar is powered by OpenStack, the leading community-driven, open source cloud infrastructure project. It seamlessly manages and provisions workloads across a heterogeneous cloud environment in a secure, compliant, and fully-supported manner. The product tightly integrates with other SUSE technologies and with the SUSE maintenance and support infrastructure.

In SUSE OpenStack Cloud Crowbar, there are several different high-level user roles:

SUSE OpenStack Cloud Operator

Installs and deploys SUSE OpenStack Cloud Crowbar on bare-metal, then installs the operating system and the OpenStack components. For detailed information about the operator's tasks and how to solve them, refer to SUSE OpenStack Cloud *Deployment Guide using Crowbar*.

SUSE OpenStack Cloud Administrator

Manages projects, users, images, flavors, and quotas within SUSE OpenStack Cloud Crowbar. For detailed information about the administrator's tasks and how to solve them, refer to the OpenStack *Administrator Guide* and the SUSE OpenStack Cloud *Supplement to Administrator Guide and User Guide*.

SUSE OpenStack Cloud User

End user who launches and manages instances, creates snapshots, and uses volumes for persistent storage within SUSE OpenStack Cloud Crowbar. For detailed information about the user's tasks and how to solve them, refer to OpenStack *User Guide* and the SUSE OpenStack Cloud *Supplement to Administrator Guide and User Guide*.

This guide provides cloud operators with the information needed to deploy and maintain SUSE OpenStack Cloud administrative units, the Administration Server, the Control Nodes, and the Compute and Storage Nodes. The Administration Server provides all services needed to manage and deploy all other nodes in the cloud. The Control Node hosts all OpenStack components needed to operate virtual machines deployed on the Compute Nodes in the SUSE OpenStack Cloud. Each virtual machine (instance) started in the cloud will be hosted on one of the Compute Nodes. Object storage is managed by the Storage Nodes.

Many chapters in this manual contain links to additional documentation resources. These include additional documentation that is available on the system, and documentation available on the Internet.

For an overview of the documentation available for your product and the latest documentation updates, refer to <https://documentation.suse.com>.

1 Available Documentation



Note: Online Documentation and Latest Updates

Documentation for our products is available at <http://documentation.suse.com>, where you can also find the latest updates, and browse or download the documentation in various formats.

In addition, the product documentation is usually available in your installed system under `/usr/share/doc/manual`. You can also access the product-specific manuals and the upstream documentation from the *Help* links in the graphical Web interfaces.

The following documentation is available for this product:

Deployment Guide using Crowbar

Gives an introduction to the SUSE® OpenStack Cloud Crowbar architecture, lists the requirements, and describes how to set up, deploy, and maintain the individual components. Also contains information about troubleshooting, support, and a glossary listing the most important terms and concepts for SUSE OpenStack Cloud Crowbar.

Administrator Guide

Introduces the OpenStack services and their components.

Also guides you through tasks like managing images, roles, instances, flavors, volumes, shares, quotas, host aggregates, and viewing cloud resources. To complete these tasks, use either the graphical Web interface (OpenStack Dashboard, code name `horizon`) or the OpenStack command line clients.

User Guide

Describes how to manage images, instances, networks, object containers, volumes, shares, stacks, and databases. To complete these tasks, use either the graphical Web interface (OpenStack Dashboard, code name `horizon`) or the OpenStack command line clients.

Supplement to Administrator Guide and User Guide

A supplement to the SUSE OpenStack Cloud Crowbar *Administrator Guide* and SUSE OpenStack Cloud Crowbar *User Guide*. It contains additional information for admins and end users that is specific to SUSE OpenStack Cloud Crowbar.

2 Feedback

Several feedback channels are available:

Services and Support Options

For services and support options available for your product, refer to <http://www.suse.com/support/>.

User Comments/Bug Reports

We want to hear your comments about and suggestions for this manual and the other documentation included with this product. If you are reading the HTML version of this guide, use the Comments feature at the bottom of each page in the online documentation at <http://documentation.suse.com>.

If you are reading the single-page HTML version of this guide, you can use the *Report Bug* link next to each section to open a bug report at <https://bugzilla.suse.com/>. A user account is needed for this.

Mail

For feedback on the documentation of this product, you can also send a mail to doc-team@suse.com. Make sure to include the document title, the product version, and the publication date of the documentation. To report errors or suggest enhancements, provide a concise description of the problem and refer to the respective section number and page (or URL).

3 Documentation Conventions

The following notices and typographical conventions are used in this documentation:



Warning

Vital information you must be aware of before proceeding. Warns you about security issues, potential loss of data, damage to hardware, or physical hazards.



Important

Important information you should be aware of before proceeding.



Note

Additional information, for example about differences in software versions.



Tip

Helpful information, like a guideline or a piece of practical advice.

- `tux > command`

Commands that can be run by any user, including the root user.

- `root # command`

Commands that must be run with root privileges. Often you can also prefix these commands with the sudo command to run them.

- /etc/passwd: directory names and file names
- PLACEHOLDER: replace PLACEHOLDER with the actual value
- PATH: the environment variable PATH
- ls, --help: commands, options, and parameters
- user: users or groups
- Alt, Alt-F1: a key to press or a key combination; keys are shown in uppercase as on a keyboard

- *File, File > Save As*: menu items, buttons
- **AMD/Intel** This paragraph is only relevant for the AMD64/Intel 64 architecture. The arrows mark the beginning and the end of the text block. ◀
- **IBM Z, POWER** This paragraph is only relevant for the architectures z Systems and POWER. The arrows mark the beginning and the end of the text block. ◀
- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.

4 About the Making of This Manual

This documentation is written in SUSEDoc, a subset of [DocBook 5](http://www.docbook.org) [\[7\]](http://www.docbook.org). The XML source files were validated by **jing**, processed by **xsltproc**, and converted into XSL-FO using a customized version of Norman Walsh's stylesheets. The final PDF is formatted through FOP from Apache Software Foundation. The open source tools and the environment used to build this documentation are provided by the DocBook Authoring and Publishing Suite (DAPS). The project's home page can be found at <https://github.com/openSUSE/daps> [\[7\]](https://github.com/openSUSE/daps).

The XML source code of this documentation can be found at <https://github.com> [\[7\]](https://github.com).

I Architecture and Requirements

- 1 The SUSE OpenStack Cloud Architecture 2
- 2 Considerations and Requirements 9

1 The SUSE OpenStack Cloud Architecture

SUSE OpenStack Cloud Crowbar is a managed cloud infrastructure solution that provides a full stack of cloud deployment and management services.

SUSE OpenStack Cloud Crowbar 9 provides the following features:

- Open source software that is based on the OpenStack Rocky release.
- Centralized resource tracking providing insight into activities and capacity of the cloud infrastructure for optimized automated deployment of services.
- A self-service portal enabling end users to configure and deploy services as necessary, and to track resource consumption (horizon).
- An image repository from which standardized, pre-configured virtual machines are published (glance).
- Automated installation processes via Crowbar using pre-defined scripts for configuring and deploying the Control Node(s) and Compute and Storage Nodes.
- Multi-tenant, role-based provisioning and access control for multiple departments and users within your organization.
- APIs enabling the integration of third-party software, such as identity management and billing solutions.
- An optional monitoring as a service solution, that allows to manage, track, and optimize the cloud infrastructure and the services provided to end users (SUSE OpenStack Cloud Crowbar Monitoring, monasca).

SUSE OpenStack Cloud Crowbar is based on SUSE Linux Enterprise Server, OpenStack, Crowbar, and Chef. SUSE Linux Enterprise Server is the underlying operating system for all cloud infrastructure machines (also called nodes). The cloud management layer, OpenStack, works as the “Cloud Operating System”. Crowbar and Chef automatically deploy and manage the OpenStack nodes from a central Administration Server.

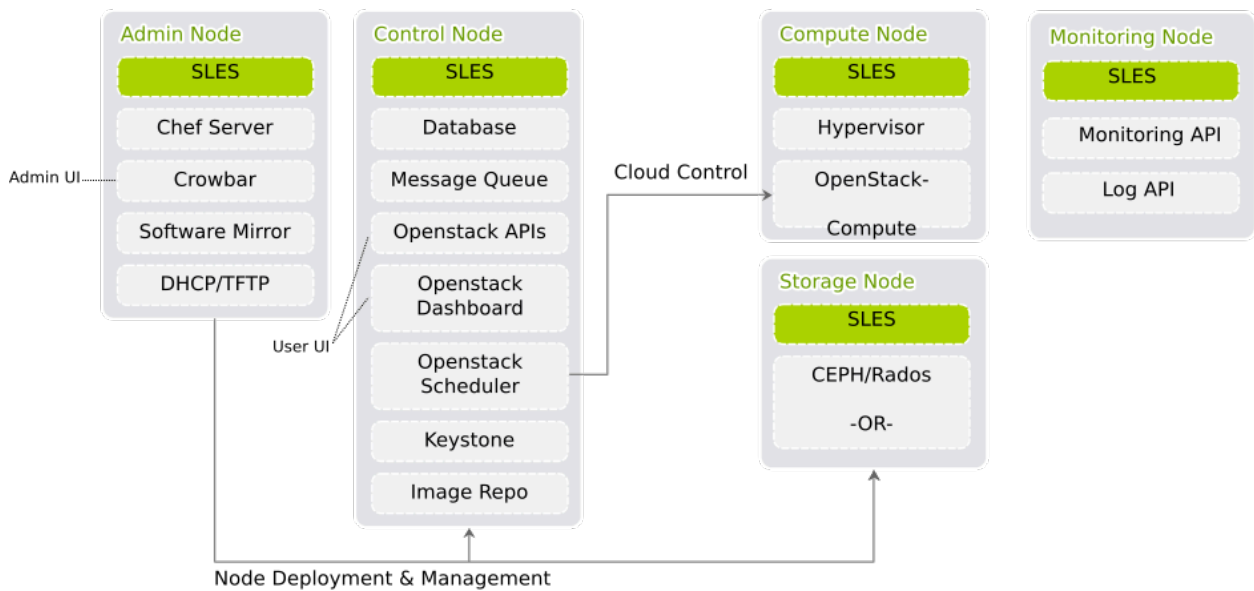


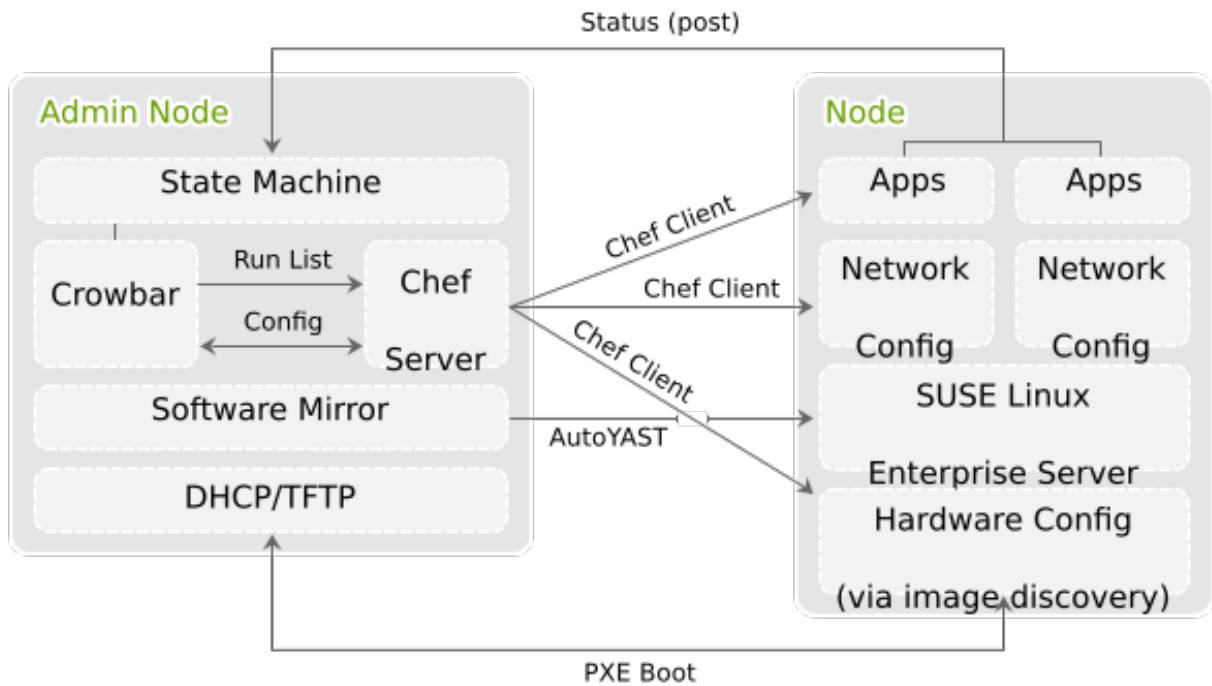
FIGURE 1.1: SUSE OPENSTACK CLOUD CROWBAR INFRASTRUCTURE

SUSE OpenStack Cloud Crowbar is deployed to four different types of machines:

- one Administration Server for node deployment and management
- one or more Control Nodes hosting the cloud management services
- several Compute Nodes on which the instances are started
- several Monitoring Node for monitoring services and servers.

1.1 The Administration Server

The Administration Server provides all services needed to manage and deploy all other nodes in the cloud. Most of these services are provided by the Crowbar tool that—together with Chef—automates all the required installation and configuration tasks. Among the services provided by the server are DHCP, DNS, NTP, PXE, and TFTP.



The Administration Server also hosts the software repositories for SUSE Linux Enterprise Server and SUSE OpenStack Cloud Crowbar, which are needed for node deployment. If no other sources for the software repositories are available, it can host the Subscription Management Tool (SMT), providing up-to-date repositories with updates and patches for all nodes.

1.2 The Control Node(s)

The Control Node(s) hosts all OpenStack components needed to orchestrate virtual machines deployed on the Compute Nodes in the SUSE OpenStack Cloud. OpenStack on SUSE OpenStack Cloud uses a MariaDB database, which is hosted on the Control Node(s). The following OpenStack components—if deployed—run on the Control Node(s):

- MariaDB database.
- Image (glance) for managing virtual images.
- Identity (keystone), providing authentication and authorization for all OpenStack components.
- Networking (neutron), providing “networking as a service” between interface devices managed by other OpenStack services.

- Block Storage (cinder), providing block storage.
- OpenStack Dashboard (horizon), providing the Dashboard, a user Web interface for the OpenStack components.
- Compute (nova) management (`nova-controller`) including API and scheduler.
- Message broker (RabbitMQ).
- swift proxy server plus dispersion tools (health monitor) and swift ring (index of objects, replicas, and devices). swift provides object storage.
- Hawk, a monitor for a pacemaker cluster in a High Availability (HA) setup.
- heat, an orchestration engine.
- designate provides DNS as a Service (DNSaaS)
- ironic, the OpenStack bare metal service for provisioning physical machines.

SUSE OpenStack Cloud Crowbar requires a three-node cluster for any production deployment since it leverages a MariaDB Galera Cluster for high availability.

We recommend deploying certain parts of Networking (neutron) on separate nodes for production clouds. See [Section 12.10, “Deploying neutron”](#) for details.

You can separate authentication and authorization services from other cloud services, for stronger security, by hosting Identity (keystone) on a separate node. Hosting Block Storage (cinder, particularly the cinder-volume role) on a separate node when using local disks for storage enables you to customize your storage and network hardware to best meet your requirements.



Note: Moving Services in an Existing Setup

If you plan to move a service from one Control Node to another, we strongly recommend shutting down or saving *all* instances before doing so. Restart them after having successfully re-deployed the services. Moving services also requires stopping them manually on the original Control Node.

1.3 The Compute Nodes

The Compute Nodes are the pool of machines on which your instances are running. These machines need to be equipped with a sufficient number of CPUs and enough RAM to start several instances. They also need to provide sufficient hard disk space, see [Section 2.2.2.3, “Compute](#)

Nodes” for details. The Control Node distributes instances within the pool of Compute Nodes and provides them with the necessary network resources. The OpenStack component Compute (nova) runs on the Compute Nodes and provides the means for setting up, starting, and stopping virtual machines.

SUSE OpenStack Cloud Crowbar supports several hypervisors, including KVM and VMware vSphere. Each image that is started with an instance is bound to one hypervisor. Each Compute Node can only run one hypervisor at a time. You will choose which hypervisor to run on each Compute Node when deploying the nova barclamp.

1.4 The Storage Nodes

The Storage Nodes are the pool of machines providing object or block storage. Object storage is provided by the OpenStack swift component, while block storage is provided by cinder. The latter supports several back-ends, including Ceph, that are deployed during the installation. Deploying swift and Ceph is optional.

1.5 The Monitoring Node

The Monitoring Node is the node that has the `monasca-server` role assigned. It hosts most services needed for SUSE OpenStack Cloud Monitoring, our monasca-based monitoring and logging solution. The following services run on this node:

Monitoring API

The monasca Web API that is used for sending metrics by monasca agents, and retrieving metrics with the monasca command line client and the monasca Grafana dashboard.

Message Queue

A Kafka instance used exclusively by SUSE OpenStack Cloud Monitoring.

Persister

Stores metrics and alarms in InfluxDB.

Notification Engine

Consumes alarms sent by the Threshold Engine and sends notifications (e.g. via email).

Threshold Engine

Based on Apache Storm. Computes thresholds on metrics and handles alarming.

Metrics and Alarms Database

A Cassandra database for storing metrics alarm history.

Config Database

A dedicated MariaDB instance used only for monitoring related data.

Log API

The monasca Web API that is used for sending log entries by monasca agents, and retrieving log entries with the Kibana Server.

Log Transformer

Transforms raw log entries sent to the Log API into a format suitable for storage.

Log Metrics

Sends metrics about high severity log messages to the Monitoring API.

Log Persister

Stores logs processed by monasca Log Transformer in the Log Database.

Kibana Server

A graphical web frontend for querying the Log Database.

Log Database

An Elasticsearch database for storing logs.

Zookeeper

Cluster synchronization for Kafka and Storm.

Currently there can only be one Monitoring node. Clustering support is planned for a future release. We strongly recommend using a dedicated physical node without any other services as a Monitoring Node.

1.6 HA Setup

A failure of components in SUSE OpenStack Cloud Crowbar can lead to system downtime and data loss. To prevent this, set up a High Availability (HA) cluster consisting of several nodes. You can assign certain roles to this cluster instead of assigning them to individual nodes. As of SUSE OpenStack Cloud Crowbar 8, Control Nodes and Compute Nodes can be made highly available.

For all HA-enabled roles, their respective functions are automatically handled by the clustering software SUSE Linux Enterprise High Availability Extension. The High Availability Extension uses the Pacemaker cluster stack with Pacemaker as cluster resource manager, and Corosync as the messaging/infrastructure layer.

View the cluster status and configuration with the cluster management tools HA Web Console (Hawk) or the `crm` shell.



Important: Do Not Change the Configuration

Use the cluster management tools only for *viewing*. All of the clustering configuration is done automatically via Crowbar and Chef. If you change anything via the cluster management tools you risk breaking the cluster. Changes done there may be reverted by the next run of Chef anyway.

A failure of the OpenStack infrastructure services (running on the Control Nodes) can be critical and may cause downtime within the cloud. For more information on making those services highly-available and avoiding other potential points of failure in your cloud setup, refer to [Section 2.6, "High Availability"](#).

2 Considerations and Requirements

Before deploying SUSE OpenStack Cloud Crowbar, there are some requirements to meet and architectural decisions to make. Read this chapter thoroughly first, as some decisions need to be made *before* deploying SUSE OpenStack Cloud, and they cannot be changed afterward.

2.1 Network

SUSE OpenStack Cloud Crowbar requires a complex network setup consisting of several networks that are configured during installation. These networks are for exclusive cloud usage. You need a router to access them from an existing network.

The network configuration on the nodes in the SUSE OpenStack Cloud network is entirely controlled by Crowbar. Any network configuration not created with Crowbar (for example, with YaST) will automatically be overwritten. After the cloud is deployed, network settings cannot be changed.

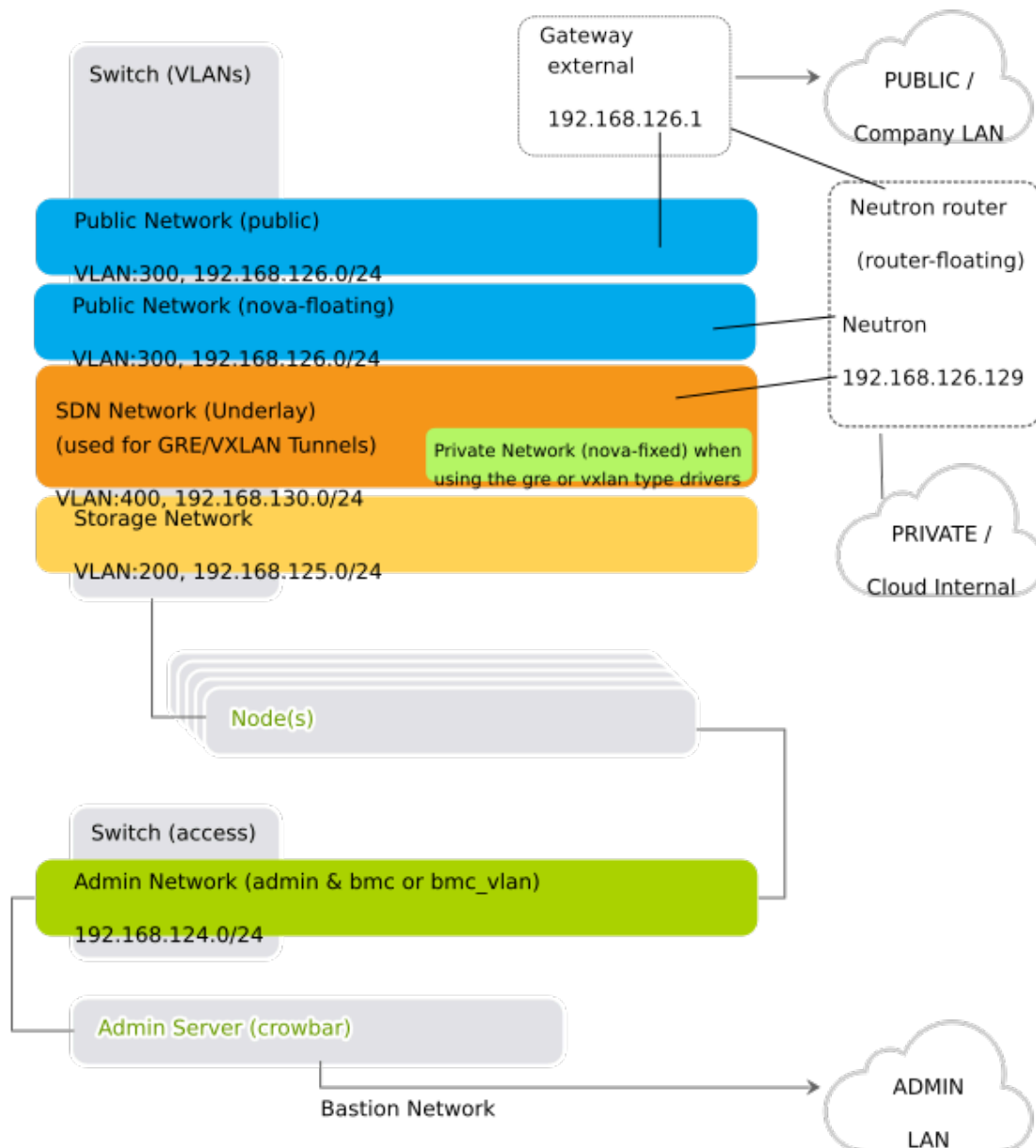


FIGURE 2.1: SUSE OPENSTACK CLOUD NETWORK: OVERVIEW

The following networks are pre-defined when setting up SUSE OpenStack Cloud Crowbar. The IP addresses listed are the default addresses and can be changed using the YaST Crowbar module (see [Chapter 7, Crowbar Setup](#)). It is also possible to customize the network setup by manually editing the network barclamp template. See [Section 7.5, "Custom Network Configuration"](#) for detailed instructions.

Admin Network (192.168.124/24)

A private network to access the Administration Server and all nodes for administration purposes. The default setup also allows access to the BMC (Baseboard Management Controller) data via IPMI (Intelligent Platform Management Interface) from this network. If required, BMC access can be swapped to a separate network.

You have the following options for controlling access to this network:

- Do not allow access from the outside and keep the admin network completely separated.
- Allow access to the Administration Server from a single network (for example, your company's administration network) via the “bastion network” option configured on an additional network card with a fixed IP address.
- Allow access from one or more networks via a gateway.

Storage Network (192.168.125/24)

Private SUSE OpenStack Cloud internal virtual network. This network is used by Ceph and swift only. It should not be accessed by users.

Private Network (nova-fixed, 192.168.123/24)

Private SUSE OpenStack Cloud internal virtual network. This network is used for inter-instance communication and provides access to the outside world for the instances. The required gateway is automatically provided by SUSE OpenStack Cloud.

Public Network (nova-floating, public, 192.168.126/24)

The only public network provided by SUSE OpenStack Cloud. You can access the nova Dashboard and all instances (provided they have been equipped with floating IP addresses) on this network. This network can only be accessed via a gateway, which must be provided externally. All SUSE OpenStack Cloud users and administrators must have access to the public network.

Software Defined Network (os_sdn, 192.168.130/24)

Private SUSE OpenStack Cloud internal virtual network. This network is used when neutron is configured to use openvswitch with GRE tunneling for the virtual networks. It should not be accessible to users.

The monasca Monitoring Network

The monasca monitoring node needs to have an interface on both the admin network and the public network. monasca's backend services will listen on the admin network, the API services (openstack-monasca-api, openstack-monasca-log-api) will listen on all

interfaces. `openstack-monasca-agent` and `openstack-monasca-log-agent` will send their logs and metrics to the `monasca-api` / `monasca-log-api` services to the monitoring node's public network IP address.

If the Octavia barclamp will be deployed, the Octavia Management Network also needs to be configured. Octavia uses a set of instances running on one or more compute nodes called amphorae and the Octavia services need to communicate with the amphorae over a dedicated management network. This network is not pre-defined when setting up SUSE OpenStack Cloud Crowbar. It needs to be explicitly configured as covered under [Section 12.20.1.1, "Management network"](#).



Warning: Protect Networks from External Access

For security reasons, protect the following networks from external access:

- *Admin Network (192.168.124/24)*
- *Storage Network (192.168.125/24)*
- *Software Defined Network (os_sdn, 192.168.130/24)*

Especially traffic from the cloud instances must not be able to pass through these networks.



Important: VLAN Settings

As of SUSE OpenStack Cloud Crowbar 8, using a VLAN for the admin network is only supported on a native/untagged VLAN. If you need VLAN support for the admin network, it must be handled at switch level.

When changing the network configuration with YaST or by editing `/etc/crowbar/network.json` you can define VLAN settings for each network. For the networks `nova-fixed` and `nova-floating`, however, special rules apply:

nova-fixed: The `USE VLAN` setting will be ignored. However, VLANs will automatically be used if deploying neutron with VLAN support (using the plugins `linuxbridge`, `openvswitch` plus `VLAN`, or `cisco` plus `VLAN`). In this case, you need to specify a correct `VLAN ID` for this network.

nova-floating: When using a VLAN for `nova-floating` (which is the default), the `USE_VLAN` and `VLAN ID` settings for `nova-floating` and `public` must be the same. When not using a VLAN for `nova-floating`, it must have a different physical network interface than the `nova_fixed` network.



Note: No IPv6 Support

As of SUSE OpenStack Cloud Crowbar 8, IPv6 is not supported. This applies to the cloud internal networks and to the instances. You must use static IPv4 addresses for all network interfaces on the Admin Node, and disable IPv6 before deploying Crowbar on the Admin Node.

The following diagram shows the pre-defined SUSE OpenStack Cloud network in more detail. It demonstrates how the OpenStack nodes and services use the different networks.

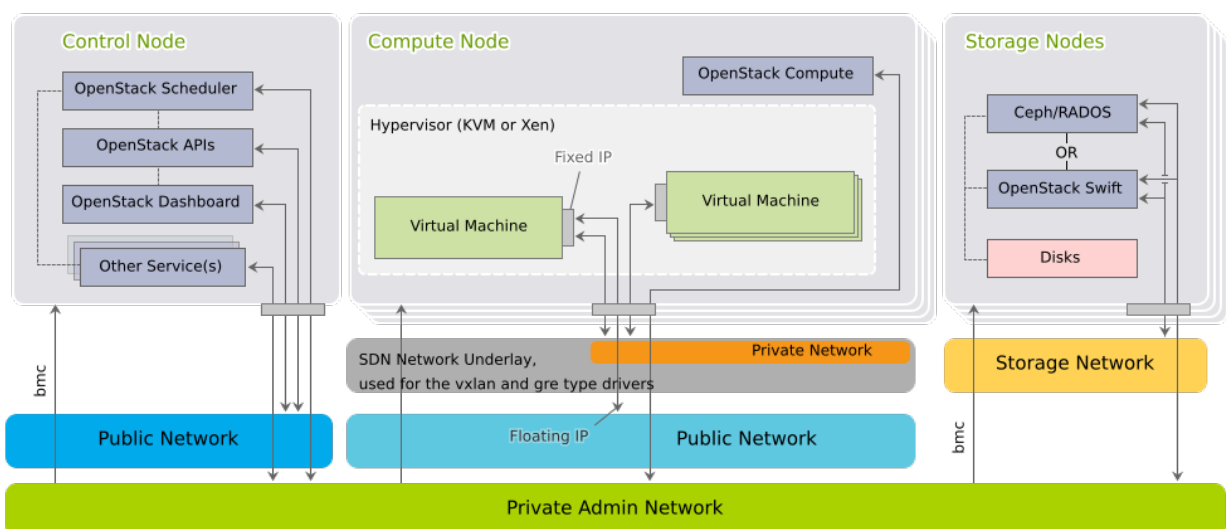


FIGURE 2.2: SUSE OPENSTACK CLOUD NETWORK: DETAILS

2.1.1 Network Address Allocation

The default networks set up in SUSE OpenStack Cloud are class C networks with 256 IP addresses each. This limits the maximum number of instances that can be started simultaneously. Addresses within the networks are allocated as outlined in the following table. Use the YaST

Crowbar module to make customizations (see [Chapter 7, Crowbar Setup](#)). The last address in the IP address range of each network is always reserved as the broadcast address. This assignment cannot be changed.

For an overview of the minimum number of IP addresses needed for each of the ranges in the network settings, see [Table 2.1, “Minimum Number of IP Addresses for Network Ranges”](#).

TABLE 2.1: MINIMUM NUMBER OF IP ADDRESSES FOR NETWORK RANGES

Network	Required Number of IP addresses
Admin Network	<ul style="list-style-type: none"> • 1 IP address per node (Administration Server, Control Nodes, and Compute Nodes) • 1 VIP address for PostgreSQL • 1 VIP address for RabbitMQ • 1 VIP address per cluster (per Pacemaker barclamp proposal)
Public Network	<ul style="list-style-type: none"> • 1 IP address per node (Control Nodes and Compute Nodes) • 1 VIP address per cluster
BMC Network	<ul style="list-style-type: none"> • 1 IP address per node (Administration Server, Control Nodes, and Compute Nodes)
Software Defined Network	<ul style="list-style-type: none"> • 1 IP address per node (Control Nodes and Compute Nodes)



Note: Limitations of the Default Network Proposal

The default network proposal as described below limits the maximum number of Compute Nodes to 80, the maximum number of floating IP addresses to 61 and the maximum number of addresses in the nova_fixed network to 204.

To overcome these limitations you need to reconfigure the network setup by using appropriate address ranges. Do this by either using the YaST Crowbar module as described in [Chapter 7, Crowbar Setup](#), or by manually editing the network template file as described in [Section 7.5, “Custom Network Configuration”](#).

TABLE 2.2: 192.168.124.0/24 (ADMIN/BMC) NETWORK ADDRESS ALLOCATION

Function	Address	Remark
router	<u>192.168.124.1</u>	Provided externally.
admin	<u>192.168.124.10</u> - <u>192.168.124.11</u>	Fixed addresses reserved for the Administration Server.
DHCP	<u>192.168.124.21</u> - <u>192.168.124.80</u>	Address range reserved for node allocation/installation. Determines the maximum number of parallel allocations/installations.
host	<u>192.168.124.81</u> - <u>192.168.124.160</u>	Fixed addresses for the OpenStack nodes. Determines the maximum number of OpenStack nodes that can be deployed.
bmc vlan host	<u>192.168.124.161</u>	Fixed address for the BMC VLAN. Used to generate a VLAN tagged interface on the Administration Server that can access the BMC network. The BMC VLAN must be in the same ranges as BMC, and BMC must have VLAN enabled.
bmc host	<u>192.168.124.162</u> - <u>192.168.124.240</u>	Fixed addresses for the OpenStack nodes. Determines the maximum number of OpenStack nodes that can be deployed.
switch	<u>192.168.124.241</u> - <u>192.168.124.250</u>	This range is not used in current releases and might be removed in the future.

TABLE 2.3: 192.168.125/24 (STORAGE) NETWORK ADDRESS ALLOCATION

Function	Address	Remark
host	<u>192.168.125.10</u> - <u>192.168.125.239</u>	Each Storage Node will get an address from this range.

TABLE 2.4: 192.168.123/24 (PRIVATE NETWORK/NOVA-FIXED) NETWORK ADDRESS ALLOCATION

Function	Address	Remark
DHCP	192.168.123.1 - 192.168.123.254	Address range for instances, routers and DHCP/DNS agents.

TABLE 2.5: 192.168.126/24 (PUBLIC NETWORK NOVA-FLOATING, PUBLIC) NETWORK ADDRESS ALLOCATION

Function	Address	Remark
router	192.168.126.1	Provided externally.
public host	192.168.126.2 - 192.168.126.127	Public address range for external SUSE OpenStack Cloud components such as the OpenStack Dashboard or the API.
floating host	192.168.126.129 - 192.168.126.254	Floating IP address range. Floating IP addresses can be manually assigned to a running instance to allow to access the guest from the outside. Determines the maximum number of instances that can concurrently be accessed from the outside. The nova_floating network is set up with a netmask of 255.255.255.192, allowing a maximum number of 61 IP addresses. This range is pre-allocated by default and managed by neutron.

TABLE 2.6: 192.168.130/24 (SOFTWARE DEFINED NETWORK) NETWORK ADDRESS ALLOCATION

Function	Address	Remark
host	192.168.130.10 - 192.168.130.254	If neutron is configured with <code>openvswitch</code> and <code>gre</code> , each network node and all Compute Nodes will get an IP address from this range.



Note: Addresses for Additional Servers

Addresses not used in the ranges mentioned above can be used to add additional servers with static addresses to SUSE OpenStack Cloud. Such servers can be used to provide additional services. A SUSE Manager server inside SUSE OpenStack Cloud, for example, must be configured using one of these addresses.

2.1.2 Network Modes

SUSE OpenStack Cloud Crowbar supports different network modes defined in Crowbar: `single`, `dual`, and `team`. As of SUSE OpenStack Cloud Crowbar 8, the networking mode is applied to all nodes and the Administration Server. That means that all machines need to meet the hardware requirements for the chosen mode. The network mode can be configured using the YaST Crowbar module (*Chapter 7, Crowbar Setup*). The network mode cannot be changed after the cloud is deployed.

Other, more flexible network mode setups can be configured by manually editing the Crowbar network configuration files. See *Section 7.5, "Custom Network Configuration"* for more information. SUSE or a partner can assist you in creating a custom setup within the scope of a consulting services agreement (see <http://www.suse.com/consulting/> for more information on SUSE consulting).



Important: Network Device Bonding is Required for HA

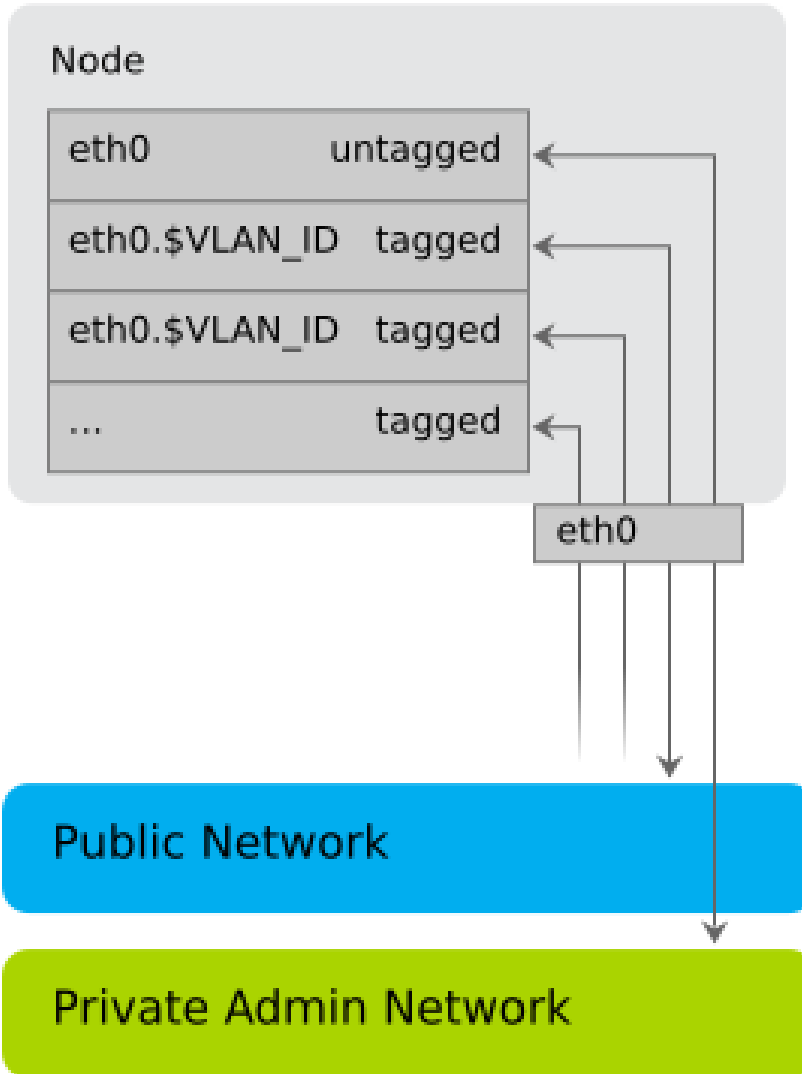
Network device bonding is required for an HA setup of SUSE OpenStack Cloud Crowbar. If you are planning to move your cloud to an HA setup at a later point in time, make sure to use a network mode in the YaST Crowbar that supports network device bonding.

Otherwise a migration to an HA setup is not supported.

2.1.2.1 Single Network Mode

In single mode you use one Ethernet card for all the traffic:

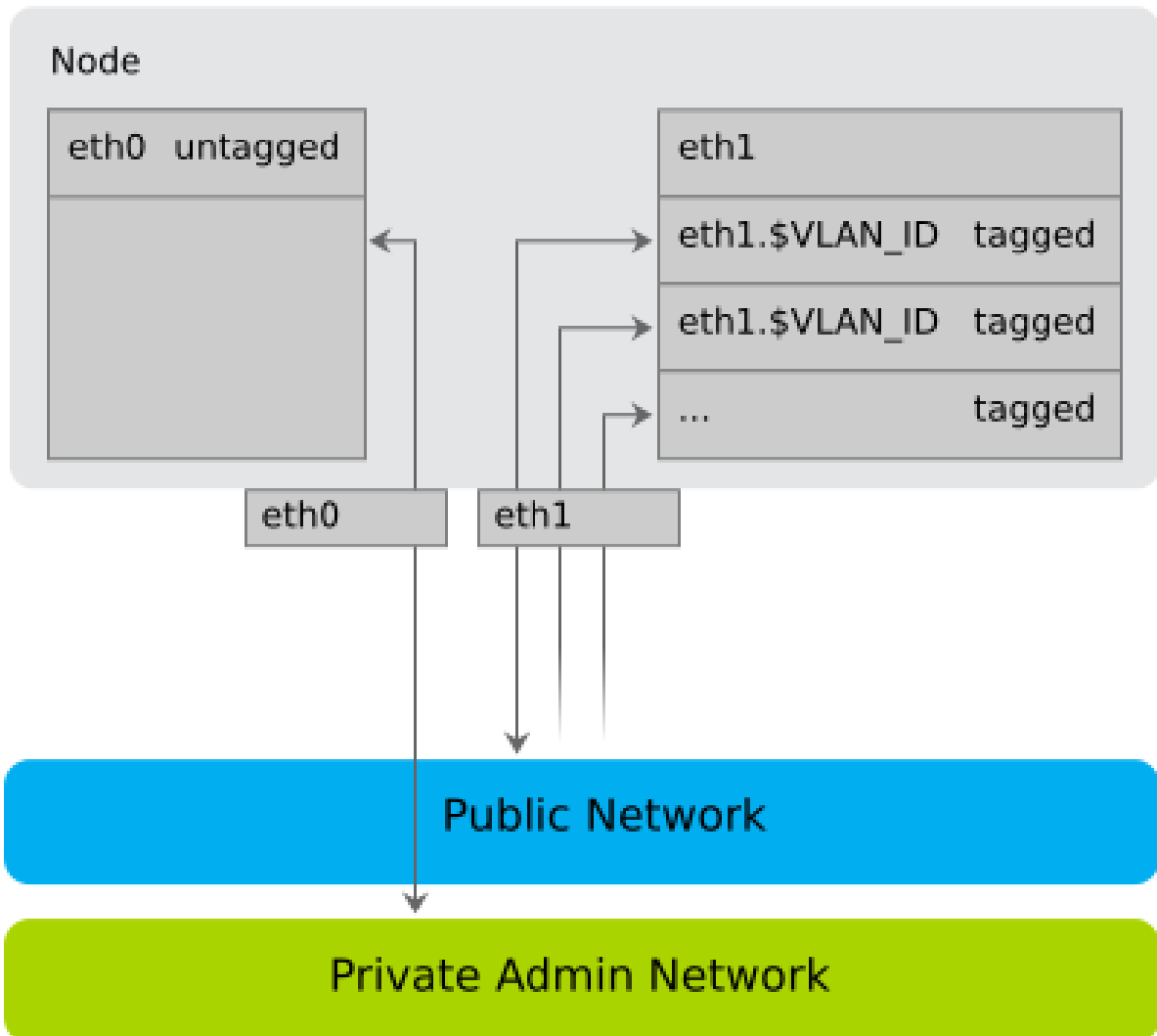
Single Mode



2.1.2.2 Dual Network Mode

Dual mode needs two Ethernet cards (on all nodes but Administration Server) to completely separate traffic between the Admin Network and the public network:

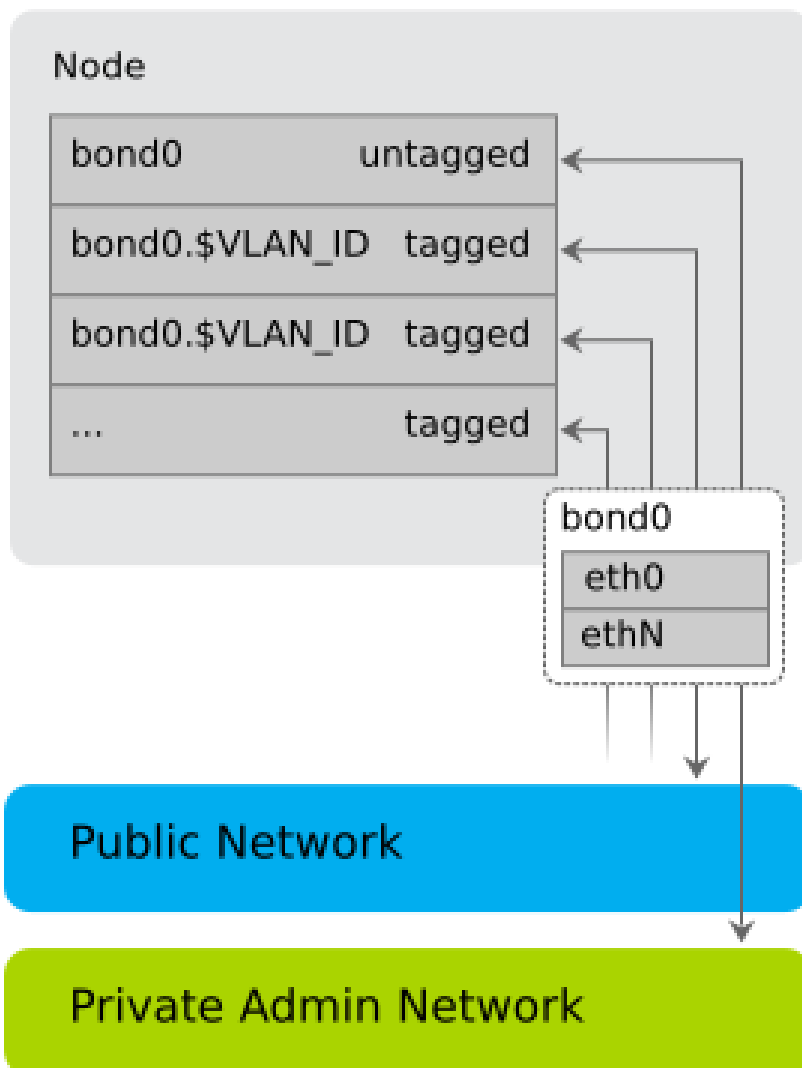
Dual Mode



2.1.2.3 Team Network Mode

Team mode is similar to single mode, except that you combine several Ethernet cards to a “bond” (network device bonding). Team mode needs two or more Ethernet cards.

Team Mode



When using team mode, you must choose a “bonding policy” that defines how to use the combined Ethernet cards. You can either set them up for fault tolerance, performance (load balancing), or a combination of both.

2.1.3 Accessing the Administration Server via a Bastion Network

Enabling access to the Administration Server from another network requires an external gateway. This option offers maximum flexibility, but requires additional hardware and may be less secure than you require. Therefore SUSE OpenStack Cloud offers a second option for accessing the Administration Server: the bastion network. You only need a dedicated Ethernet card and a static IP address from the external network to set it up.

The bastion network setup (see [Section 7.3.1, “Setting Up a Bastion Network”](#) for setup instructions) enables logging in to the Administration Server via SSH from the company network. A direct login to other nodes in the cloud is not possible. However, the Administration Server can act as a “jump host”: First log in to the Administration Server via SSH, then log in via SSH to other nodes.

2.1.4 DNS and Host Names

The Administration Server acts as a name server for all nodes in the cloud. If the Administration Server has access to the outside, then you can add additional name servers that are automatically used to forward requests. If additional name servers are found on your cloud deployment, the name server on the Administration Server is automatically configured to forward requests for non-local records to these servers.

The Administration Server must have a fully qualified host name. The domain name you specify is used for the DNS zone. It is required to use a sub-domain such as `cloud.example.com`. The Administration Server must have authority over the domain it is on so that it can create records for discovered nodes. As a result, it will not forward requests for names it cannot resolve in this domain, and thus cannot resolve names for the second-level domain, .e.g. `example.com`, other than for nodes in the cloud.

This host name must not be changed after SUSE OpenStack Cloud has been deployed. The OpenStack nodes are named after their MAC address by default, but you can provide aliases, which are easier to remember when allocating the nodes. The aliases for the OpenStack nodes can be changed at any time. It is useful to have a list of MAC addresses and the intended use of the corresponding host at hand when deploying the OpenStack nodes.

2.2 Persistent Storage

When talking about “persistent storage” on SUSE OpenStack Cloud Crowbar, there are two completely different aspects to discuss: 1) the block and object storage services SUSE OpenStack Cloud offers, 2) the hardware related storage aspects on the different node types.



Note: Persistent vs. Ephemeral Storage

Block and object storage are persistent storage models where files or images are stored until they are explicitly deleted. SUSE OpenStack Cloud also offers ephemeral storage for images attached to instances. These ephemeral images only exist during the life of an instance and are deleted when the guest is terminated. See [Section 2.2.2.3, “Compute Nodes”](#) for more information.

2.2.1 Cloud Storage Services

SUSE OpenStack Cloud offers two different types of services for persistent storage: object and block storage. Object storage lets you upload and download files (similar to an FTP server), whereas a block storage provides mountable devices (similar to a hard disk partition). SUSE OpenStack Cloud provides a repository to store the virtual disk images used to start instances.

Object Storage with swift

The OpenStack object storage service is called swift. The storage component of swift (swift-storage) must be deployed on dedicated nodes where no other cloud services run. Deploy at least two swift nodes to provide redundant storage. SUSE OpenStack Cloud Crowbar is configured to always use all unused disks on a node for storage.

swift can optionally be used by glance, the service that manages the images used to boot the instances. Offering object storage with swift is optional.

Block Storage

Block storage on SUSE OpenStack Cloud is provided by cinder. cinder can use a variety of storage back-ends, including network storage solutions like NetApp or EMC. It is also possible to use local disks for block storage. A list of drivers available for cinder and the features supported for each driver is available from the *CinderSupportMatrix* at <https://wiki.openstack.org/wiki/CinderSupportMatrix>. SUSE OpenStack Cloud Crowbar 9 ships with OpenStack Rocky.

Alternatively, cinder can use Ceph RBD as a back-end. Ceph offers data security and speed by storing the the content on a dedicated Ceph cluster.

The glance Image Repository

glance provides a catalog and repository for virtual disk images used to start the instances. glance is installed on a Control Node. It uses swift, Ceph, or a directory on the Control Node to store the images. The image directory can either be a local directory or an NFS share.

2.2.2 Storage Hardware Requirements

Each node in SUSE OpenStack Cloud needs sufficient disk space to store both the operating system and additional data. Requirements and recommendations for the various node types are listed below.



Important: Choose a Hard Disk for the Operating System Installation

The operating system will always be installed on the *first* hard disk. This is the disk that is listed *first* in the BIOS, the one from which the machine will boot. Make sure that the hard disk the operating system is installed on will be recognized as the first disk.

2.2.2.1 Administration Server

If you store the update repositories directly on the Administration Server (see [Section 2.5.2, “Product and Update Repositories”](#)), we recommend mounting `/srv` on a separate partition or volume with a minimum of 30 GB space.

Log files from all nodes in SUSE OpenStack Cloud are stored on the Administration Server under `/var/log`. The message service RabbitMQ requires 1 GB of free space in `/var` (see *Book “Operations Guide Crowbar”, Chapter 5 “Log Management”, Section 5.4 “Log Files”, Section 5.4.1 “On the Administration Server”* for a complete list).

2.2.2.2 Control Nodes

Depending on how the services are set up, glance and cinder may require additional disk space on the Control Node on which they are running. glance may be configured to use a local directory, whereas cinder may use a local image file for storage. For performance and scalability reasons this is only recommended for test setups. Make sure there is sufficient free disk space available if you use a local file for storage.

cinder may be configured to use local disks for storage (configuration option `raw`). If you choose this setup, we recommend deploying the `cinder-volume` role to one or more dedicated Control Nodes. Those should be equipped with several disks providing sufficient storage space. It may also be necessary to equip this node with two or more bonded network cards, since it will generate heavy network traffic. Bonded network cards require a special setup for this node. For details, refer to [Section 7.5, “Custom Network Configuration”](#).

2.2.2.3 Compute Nodes

Unless an instance is started via “Boot from Volume”, it is started with at least one disk, which is a copy of the image from which it has been started. Depending on the flavor you start, the instance may also have a second, so-called “ephemeral” disk. The size of the root disk depends on the image itself. Ephemeral disks are always created as sparse image files that grow up to a defined size as they are “filled”. By default ephemeral disks have a size of 10 GB.

Both disks, root images and ephemeral disk, are directly bound to the instance and are deleted when the instance is terminated. These disks are bound to the Compute Node on which the instance has been started. The disks are created under `/var/lib/nova` on the Compute Node. Your Compute Nodes should be equipped with enough disk space to store the root images and ephemeral disks.



Note: Ephemeral Disks vs. Block Storage

Do not confuse ephemeral disks with persistent block storage. In addition to an ephemeral disk, which is automatically provided with most instance flavors, you can optionally add a persistent storage device provided by cinder. Ephemeral disks are deleted when the instance terminates, while persistent storage devices can be reused in another instance.

The maximum disk space required on a compute node depends on the available flavors. A flavor specifies the number of CPUs, RAM, and disk size of an instance. Several flavors ranging from *tiny* (1 CPU, 512 MB RAM, no ephemeral disk) to *xlarge* (8 CPUs, 8 GB RAM, 10 GB ephemeral disk) are available by default. Adding custom flavors, and editing and deleting existing flavors is also supported.

To calculate the minimum disk space needed on a compute node, you need to determine the highest disk-space-to-RAM ratio from your flavors. For example:

Flavor small: 2 GB RAM, 100 GB ephemeral disk = > 50 GB disk /1 GB RAM

Flavor large: 8 GB RAM, 200 GB ephemeral disk = > 25 GB disk /1 GB RAM

So, 50 GB disk /1 GB RAM is the ratio that matters. If you multiply that value by the amount of RAM in GB available on your compute node, you have the minimum disk space required by ephemeral disks. Pad that value with sufficient space for the root disks plus a buffer to leave room for flavors with a higher disk-space-to-RAM ratio in the future.



Warning: Overcommitting Disk Space

The scheduler that decides in which node an instance is started does not check for available disk space. If there is no disk space left on a compute node, this will not only cause data loss on the instances, but the compute node itself will also stop operating. Therefore you must make sure all compute nodes are equipped with enough hard disk space.

2.2.2.4 Storage Nodes (optional)

The block storage service Ceph RBD and the object storage service swift need to be deployed onto dedicated nodes—it is not possible to mix these services. The swift component requires at least two machines (more are recommended) to store data redundantly. For information on hardware requirements for Ceph, see <https://documentation.suse.com/ses/5.5/single-html/ses-deployment/#storage-bp-hwreq>

Each Ceph/swift Storage Node needs at least two hard disks. The first one will be used for the operating system installation, while the others can be used for storage. We recommend equipping the storage nodes with as many disks as possible.

Using RAID on swift storage nodes is not supported. swift takes care of redundancy and replication on its own. Using RAID with swift would also result in a huge performance penalty.

2.3 SSL Encryption

Whenever non-public data travels over a network it must be encrypted. Encryption protects the integrity and confidentiality of data. Therefore you should enable SSL support when deploying SUSE OpenStack Cloud to production. (SSL is not enabled by default as it requires you to provide certificates.) The following services (and their APIs, if available) can use SSL:

- cinder
- horizon
- glance
- heat
- keystone
- manila
- neutron
- nova
- swift
- VNC
- RabbitMQ
- ironic
- Magnum

You have two options for deploying your SSL certificates. You may use a single shared certificate for all services on each node, or provide individual certificates for each service. The minimum requirement is a single certificate for the Control Node and all services installed on it.

Certificates must be signed by a trusted authority. Refer to <https://documentation.suse.com/sles/15-SP1/single-html/SLES-admin/#sec-apache2-ssl> for instructions on how to create and sign them.

! Important: Host Names

Each SSL certificate is issued for a certain host name and, optionally, for alternative host names (via the `AlternativeName` option). Each publicly available node in SUSE OpenStack Cloud has two host names—an internal and a public one. The SSL certificate needs to be issued for *both* internal and public names.

The internal name has the following scheme:

```
dMACADDRESS.FQDN
```

`MACADDRESS` is the MAC address of the interface used to boot the machine via PXE. All letters are turned lowercase and all colons are replaced with dashes. For example, `00-00-5E-00-53-00`. `FQDN` is the fully qualified domain name. An example name looks like this:

```
d00-00-5E-00-53-00.example.com
```

Unless you have entered a custom *Public Name* for a client (see [Section 11.2, “Node Installation”](#) for details), the public name is the same as the internal name prefixed by `public`:

```
public-d00-00-5E-00-53-00.example.com
```

To look up the node names open the Crowbar Web interface and click the name of a node in the *Node Dashboard*. The names are listed as *Full Name* and *Public Name*.

2.4 Hardware Requirements

Precise hardware requirements can only be listed for the Administration Server and the OpenStack Control Node. The requirements of the OpenStack Compute and Storage Nodes depends on the number of concurrent instances and their virtual hardware equipment.

A minimum of three machines are required for a SUSE OpenStack Cloud: one Administration Server, one Control Node, and one Compute Node. You also need a gateway providing access to the public network. Deploying storage requires additional nodes: at least two nodes for swift and a minimum of four nodes for Ceph.



Important: Virtual/Physical Machines and Architecture

Deploying SUSE OpenStack Cloud functions to virtual machines is only supported for the Administration Server—all other nodes need to be physical hardware. Although the Control Node can be virtualized in test environments, this is not supported for production systems.

SUSE OpenStack Cloud currently only runs on `x86_64` hardware.

2.4.1 Administration Server

- Architecture: `x86_64`.
- RAM: at least 4 GB, 8 GB recommended. The demand for memory depends on the total number of nodes in SUSE OpenStack Cloud—the higher the number of nodes, the more RAM is needed. A deployment with 50 nodes requires a minimum of 24 GB RAM for each Control Node.
- Hard disk: at least 50 GB. We recommend putting `/srv` on a separate partition with at least additional 30 GB of space. Alternatively, you can mount the update repositories from another server (see [Section 2.5.2, “Product and Update Repositories”](#) for details).
- Number of network cards: 1 for single and dual mode, 2 or more for team mode. Additional networks such as the bastion network and/or a separate BMC network each need an additional network card. See [Section 2.1, “Network”](#) for details.
- Can be deployed on physical hardware or a virtual machine.

2.4.2 Control Node

- Architecture: `x86_64`.
- RAM: at least 8 GB, 12 GB when deploying a single Control Node, and 32 GB recommended.
- Number of network cards: 1 for single mode, 2 for dual mode, 2 or more for team mode. See [Section 2.1, “Network”](#) for details.
- Hard disk: See [Section 2.2.2.2, “Control Nodes”](#).

2.4.3 Compute Node

The Compute Nodes need to be equipped with a sufficient amount of RAM and CPUs, matching the numbers required by the maximum number of instances running concurrently. An instance started in SUSE OpenStack Cloud cannot share resources from several physical nodes. It uses the resources of the node on which it was started. So if you offer a flavor (see *Flavor* for a definition) with 8 CPUs and 12 GB RAM, at least one of your nodes should be able to provide these resources. Add 1 GB RAM for every two nodes (including Control Nodes and Storage Nodes) deployed in your cloud.

See *Section 2.2.2.3, "Compute Nodes"* for storage requirements.

2.4.4 Storage Node

Usually a single CPU and a minimum of 4 GB RAM are sufficient for the Storage Nodes. Memory requirements increase depending on the total number of nodes in SUSE OpenStack Cloud—the higher the number of nodes, the more RAM you need. A deployment with 50 nodes requires a minimum of 20 GB for each Storage Node. If you use Ceph as storage, the storage nodes should be equipped with an additional 2 GB RAM per OSD (Ceph object storage daemon).

For storage requirements, see *Section 2.2.2.4, "Storage Nodes (optional)"*.

2.4.5 monasca Node

The monasca Node is a dedicated physical machine that runs the `monasca-server` role. This node is used for SUSE OpenStack Cloud Crowbar Monitoring. Hardware requirements for the monasca Node are as follows:

- Architecture: x86_64
- RAM: At least 32 GB, 64 GB or more is recommended
- CPU: At least 8 cores, 16 cores or more is recommended
- Hard Disk: SSD is strongly recommended

The following formula can be used to calculate the required disk space:

```
200 GB + ["number of nodes" * "retention period" * ("space for log  
data/day" + "space for metrics data/day") ]
```

The recommended values for the formula are as follows:

- Retention period = 60 days for InfluxDB and Elasticsearch
- Space for daily log data = 2GB
- Space for daily metrics data = 50MB

The formula is based on the following log data assumptions:

- Approximately 50 log files per node
- Approximately 1 log entry per file per sec
- 200 bytes in size

The formula is based on the following metrics data assumptions:

- 400 metrics per node
- Time interval of 30 seconds
- 20 bytes in size

The formula provides only a rough estimation of the required disk space. There are several factors that can affect disk space requirements. This includes the exact combination of services that run on your OpenStack node actual cloud usage pattern, and whether any or all services have debug logging enabled.

2.5 Software Requirements

All nodes and the Administration Server in SUSE OpenStack Cloud run on SUSE Linux Enterprise Server 12 SP4. Subscriptions for the following components are available as one- or three-year subscriptions including priority support:

- SUSE OpenStack Cloud Crowbar Control Node + SUSE OpenStack Cloud Crowbar Administration Server (including entitlements for High Availability and SUSE Linux Enterprise Server 12 SP4)
- Additional SUSE OpenStack Cloud Crowbar Control Node (including entitlements for High Availability and SUSE Linux Enterprise Server 12 SP4)

- SUSE OpenStack Cloud Crowbar Compute Node (excluding entitlements for High Availability and SUSE Linux Enterprise Server 12 SP4)
- SUSE OpenStack Cloud Crowbar swift node (excluding entitlements for High Availability and SUSE Linux Enterprise Server 12 SP4)

SUSE Linux Enterprise Server 12 SP4, HA entitlements for Compute Nodes and swift Storage Nodes, and entitlements for guest operating systems need to be purchased separately. Refer to <http://www.suse.com/products/suse-openstack-cloud/how-to-buy/> for more information on licensing and pricing.

Running an external Ceph cluster (optional) with SUSE OpenStack Cloud requires an additional SUSE Enterprise Storage subscription. Refer to <https://www.suse.com/products/suse-enterprise-storage/> and <https://www.suse.com/products/suse-openstack-cloud/frequently-asked-questions> for more information.



Important: SUSE Account

A SUSE account is needed for product registration and access to update repositories. If you do not already have one, go to <http://www.suse.com/> to create it.

2.5.1 Optional Component: SUSE Enterprise Storage

SUSE OpenStack Cloud Crowbar can be extended by SUSE Enterprise Storage for setting up a Ceph cluster providing block storage services. To store virtual disks for instances, SUSE OpenStack Cloud uses block storage provided by the cinder module. cinder itself needs a back-end providing storage. In production environments this usually is a network storage solution. cinder can use a variety of network storage back-ends, among them solutions from EMC, Fujitsu, or NetApp. In case your organization does not provide a network storage solution that can be used with SUSE OpenStack Cloud, you can set up a Ceph cluster with SUSE Enterprise Storage. SUSE Enterprise Storage provides a reliable and fast distributed storage architecture using commodity hardware platforms.

2.5.2 Product and Update Repositories

You need seven software repositories to deploy SUSE OpenStack Cloud and to keep a running SUSE OpenStack Cloud up-to-date. This includes the static product repositories, which do not change over the product life cycle, and the update repositories, which constantly change. The following repositories are needed:

MANDATORY REPOSITORIES

SUSE Linux Enterprise Server 12 SP4 Product

The SUSE Linux Enterprise Server 12 SP4 product repository is a copy of the installation media (DVD #1) for SUSE Linux Enterprise Server. As of SUSE OpenStack Cloud Crowbar 8, it is required to have it available locally on the Administration Server. This repository requires approximately 3.5 GB of hard disk space.

SUSE OpenStack Cloud Crowbar 9 Product

The SUSE OpenStack Cloud Crowbar 9 product repository is a copy of the installation media (DVD #1) for SUSE OpenStack Cloud. It can either be made available remotely via HTTP, or locally on the Administration Server. We recommend the latter since it makes the setup of the Administration Server easier. This repository requires approximately 500 MB of hard disk space.

PTF

This repository is created automatically on the Administration Server when you install the SUSE OpenStack Cloud add-on product. It serves as a repository for “Program Temporary Fixes” (PTF), which are part of the SUSE support program.

SLES12-SP4-Pool and SUSE-OpenStack-Cloud-Crowbar-9-Pool

The SUSE Linux Enterprise Server and SUSE OpenStack Cloud Crowbar repositories contain all binary RPMs from the installation media, plus pattern information and support status metadata. These repositories are served from SUSE Customer Center and need to be kept in synchronization with their sources. Make them available remotely via an existing SMT or SUSE Manager server. Alternatively, make them available locally on the Administration Server by installing a local SMT server, by mounting or synchronizing a remote directory, or by copying them.

SLES12-SP4-Updates and SUSE-OpenStack-Cloud-Crowbar-9-Updates

These repositories contain maintenance updates to packages in the corresponding Pool repositories. These repositories are served from SUSE Customer Center and need to be kept synchronized with their sources. Make them available remotely via an existing SMT or SUSE Manager server, or locally on the Administration Server by installing a local SMT server, by mounting or synchronizing a remote directory, or by regularly copying them.

As explained in [Section 2.6, “High Availability”](#), Control Nodes in SUSE OpenStack Cloud can optionally be made highly available with the SUSE Linux Enterprise High Availability Extension. The following repositories are required to deploy SLES High Availability Extension nodes:

OPTIONAL REPOSITORIES

SLE-HA12-SP4-Pool

The pool repositories contain all binary RPMs from the installation media, plus pattern information and support status metadata. These repositories are served from SUSE Customer Center and need to be kept in synchronization with their sources. Make them available remotely via an existing SMT or SUSE Manager server. Alternatively, make them available locally on the Administration Server by installing a local SMT server, by mounting or synchronizing a remote directory, or by copying them.

SLE-HA12-SP4-Updates

These repositories contain maintenance updates to packages in the corresponding pool repositories. These repositories are served from SUSE Customer Center and need to be kept synchronized with their sources. Make them available remotely via an existing SMT or SUSE Manager server, or locally on the Administration Server by installing a local SMT server, by mounting or synchronizing a remote directory, or by regularly copying them.

The product repositories for SUSE Linux Enterprise Server 12 SP4 and SUSE OpenStack Cloud Crowbar 9 do not change during the life cycle of a product. Thus, they can be copied to the destination directory from the installation media. However, the pool and update repositories must be kept synchronized with their sources on the SUSE Customer Center. SUSE offers two products that synchronize repositories and make them available within your organization: SUSE Manager (<http://www.suse.com/products/suse-manager/>), and Subscription Management Tool (which ships with SUSE Linux Enterprise Server 12 SP4).

All repositories must be served via HTTP to be available for SUSE OpenStack Cloud deployment. Repositories that are installed on the Administration Server are made available by the Apache Web server running on the Administration Server. If your organization already uses SUSE Manager or SMT, you can use the repositories provided by these servers.

Making the repositories locally available on the Administration Server has the advantage of a simple network setup within SUSE OpenStack Cloud, and it allows you to seal off the SUSE OpenStack Cloud network from other networks in your organization. Hosting the repositories on a remote server has the advantage of using existing resources and services, and it makes setting up the Administration Server much easier. However, this requires a custom network setup for SUSE OpenStack Cloud, since the Administration Server needs access to the remote server.

Installing a Subscription Management Tool (SMT) Server on the Administration Server

The SMT server, shipping with SUSE Linux Enterprise Server 12 SP4, regularly synchronizes repository data from SUSE Customer Center with your local host. Installing the SMT server on the Administration Server is recommended if you do not have access to update repositories from elsewhere within your organization. This option requires the Administration Server to have Internet access.

Using a Remote SMT Server

If you already run an SMT server within your organization, you can use it within SUSE OpenStack Cloud. When using a remote SMT server, update repositories are served directly from the SMT server. Each node is configured with these repositories upon its initial setup. The SMT server needs to be accessible from the Administration Server and all nodes in SUSE OpenStack Cloud (via one or more gateways). Resolving the server's host name also needs to work.

Using a SUSE Manager Server

Each client that is managed by SUSE Manager needs to register with the SUSE Manager server. Therefore the SUSE Manager support can only be installed after the nodes have been deployed. SUSE Linux Enterprise Server 12 SP4 must be set up for autoinstallation on the SUSE Manager server in order to use repositories provided by SUSE Manager during node deployment.

The server needs to be accessible from the Administration Server and all nodes in SUSE OpenStack Cloud (via one or more gateways). Resolving the server's host name also needs to work.

Using Existing Repositories

If you can access existing repositories from within your company network from the Administration Server, you have the following options: mount, synchronize, or manually transfer these repositories to the required locations on the Administration Server.

2.6 High Availability

Several components and services in SUSE OpenStack Cloud Crowbar are potentially single points of failure that may cause system downtime and data loss if they fail.

SUSE OpenStack Cloud Crowbar provides various mechanisms to ensure that the crucial components and services are highly available. The following sections provide an overview of components on each node that can be made highly available. For making the Control Node functions and the Compute Nodes highly available, SUSE OpenStack Cloud Crowbar uses the cluster software SUSE Linux Enterprise High Availability Extension. Make sure to thoroughly read [Section 2.6.5, “Cluster Requirements and Recommendations”](#) to learn about additional requirements for high availability deployments.

2.6.1 High Availability of the Administration Server

The Administration Server provides all services needed to manage and deploy all other nodes in the cloud. If the Administration Server is not available, new cloud nodes cannot be allocated, and you cannot add new roles to cloud nodes.

However, only two services on the Administration Server are single points of failure, without which the cloud cannot continue to run properly: DNS and NTP.

2.6.1.1 Administration Server—Avoiding Points of Failure

To avoid DNS and NTP as potential points of failure, deploy the roles `dns-server` and `ntp-server` to multiple nodes.



Note: Access to External Network

If any configured DNS forwarder or NTP external server is not reachable through the admin network from these nodes, allocate an address in the public network for each node that has the `dns-server` and `ntp-server` roles:

```
crowbar network allocate_ip default `hostname -f` public host
```

Then the nodes can use the public gateway to reach the external servers. The change will only become effective after the next run of `chef-client` on the affected nodes.

2.6.2 High Availability of the Control Node(s)

The Control Node(s) usually run a variety of services without which the cloud would not be able to run properly.

2.6.2.1 Control Node(s)—Avoiding Points of Failure

To prevent the cloud from avoidable downtime if one or more Control Nodes fail, you can make the following roles highly available:

- [database-server](#) ([database](#) barclamp)
- [keystone-server](#) ([keystone](#) barclamp)
- [rabbitmq-server](#) ([rabbitmq](#) barclamp)
- [swift-proxy](#) ([swift](#) barclamp)
- [glance-server](#) ([glance](#) barclamp)
- [cinder-controller](#) ([cinder](#) barclamp)
- [neutron-server](#) ([neutron](#) barclamp)
- [neutron-network](#) ([neutron](#) barclamp)
- [nova-controller](#) ([nova](#) barclamp)
- [nova_dashboard-server](#) ([nova_dashboard](#) barclamp)
- [ceilometer-server](#) ([ceilometer](#) barclamp)
- [ceilometer-control](#) ([ceilometer](#) barclamp)
- [heat-server](#) ([heat](#) barclamp)
- [octavia-api](#) ([octavia](#) barclamp)
- [octavia-backend](#) ([octavia](#) barclamp)

Instead of assigning these roles to individual cloud nodes, you can assign them to one or several High Availability clusters. SUSE OpenStack Cloud Crowbar will then use the Pacemaker cluster stack (shipped with the SUSE Linux Enterprise High Availability Extension) to manage the services. If one Control Node fails, the services will fail over to another Control Node. For details on the Pacemaker cluster stack and the SUSE Linux Enterprise High Availability Extension, refer to the *High Availability Guide*, available at <https://documentation.suse.com/sle-ha/15-SP1/>.

Note that SUSE Linux Enterprise High Availability Extension includes Linux Virtual Server as the load-balancer, and SUSE OpenStack Cloud Crowbar uses HAProxy for this purpose (<http://haproxy.1wt.eu/>).



Note: Recommended Setup

Though it is possible to use the same cluster for all of the roles above, the recommended setup is to use three clusters and to deploy the roles as follows:

- data cluster: database-server and rabbitmq-server
- network cluster: neutron-network (as the neutron-network role may result in heavy network load and CPU impact)
- services cluster: all other roles listed above (as they are related to API/schedulers)



Important: Cluster Requirements and Recommendations

For setting up the clusters, some special requirements and recommendations apply. For details, refer to [Section 2.6.5, “Cluster Requirements and Recommendations”](#).

2.6.2.2 Control Node(s)—Recovery

Recovery of the Control Node(s) is done automatically by the cluster software: if one Control Node fails, Pacemaker will fail over the services to another Control Node. If a failed Control Node is repaired and rebuilt via Crowbar, it will be automatically configured to join the cluster. At this point Pacemaker will have the option to fail back services if required.

2.6.3 High Availability of the Compute Node(s)

If a Compute Node fails, all VMs running on that node will go down. While it cannot protect against failures of individual VMs, a High Availability setup for Compute Nodes helps to minimize VM downtime caused by Compute Node failures. If the nova-compute service or libvirt fail on a Compute Node, Pacemaker will try to automatically recover them. If recovery fails, or the node itself should become unreachable, the node will be fenced and the VMs will be moved to a different Compute Node.

If you decide to use High Availability for Compute Nodes, your Compute Node will be run as Pacemaker remote nodes. With the `pacemaker-remote` service, High Availability clusters can be extended to control remote nodes without any impact on scalability, and without having to install the full cluster stack (including `corosync`) on the remote nodes. Instead, each Compute Node only runs the `pacemaker-remote` service. The service acts as a proxy, allowing the cluster stack on the “normal” cluster nodes to connect to it and to control services remotely. Thus, the node is effectively integrated into the cluster as a remote node. In this way, the services running on the OpenStack compute nodes can be controlled from the core Pacemaker cluster in a lightweight, scalable fashion.

Find more information about the `pacemaker_remote` service in *Pacemaker Remote—Extending High Availability into Virtual Nodes*, available at <http://www.clusterlabs.org/doc/>.

To configure High Availability for Compute Nodes, you need to adjust the following barclamp proposals:

- Pacemaker—for details, see *Section 12.2, “Deploying Pacemaker (Optional, HA Setup Only)”*.
- nova—for details, see *Section 12.11.1, “HA Setup for nova”*.

2.6.4 High Availability of the Storage Node(s)

SUSE OpenStack Cloud Crowbar offers two different types of storage that can be used for the Storage Nodes: object storage (provided by the OpenStack swift component) and block storage (provided by Ceph).

Both already consider High Availability aspects by design, therefore it does not require much effort to make the storage highly available.

2.6.4.1 swift—Avoiding Points of Failure

The OpenStack Object Storage replicates the data by design, provided the following requirements are met:

- The option `Replicas` in the swift barclamp is set to `3`, the tested and recommended value.
- The number of Storage Nodes needs to be greater than the value set in the `Replicas` option.

1. To avoid single points of failure, assign the `swift-storage` role to multiple nodes.

2. To make the API highly available, assign the `swift-proxy` role to a cluster instead of assigning it to a single Control Node. See [Section 2.6.2.1, “Control Node\(s\)—Avoiding Points of Failure”](#). Other swift roles must not be deployed on a cluster.

2.6.4.2 Ceph—Avoiding Points of Failure

Ceph is a distributed storage solution that can provide High Availability. For High Availability redundant storage and monitors need to be configured in the Ceph cluster. For more information refer to the SUSE Enterprise Storage documentation at <https://documentation.suse.com/ses/5.5/>.

2.6.5 Cluster Requirements and Recommendations

When considering setting up one or more High Availability clusters, refer to the chapter *System Requirements* in the *High Availability Guide* for SUSE Linux Enterprise High Availability Extension. The guide is available at <https://documentation.suse.com/sle-ha/15-SP1/>.

The HA requirements for Control Node also apply to SUSE OpenStack Cloud Crowbar. Note that by buying SUSE OpenStack Cloud Crowbar, you automatically get an entitlement for SUSE Linux Enterprise High Availability Extension.

Especially note the following requirements:

Number of Cluster Nodes

Each cluster needs to consist of at least three cluster nodes.



Important: Odd Number of Cluster Nodes

The Galera cluster needs an *odd* number of cluster nodes with a *minimum* of three nodes.

A cluster needs *Quorum* to keep services running. A three-node cluster can tolerate failure of only one node at a time, whereas a five-node cluster can tolerate failures of two nodes.

STONITH

The cluster software will shut down “misbehaving” nodes in a cluster to prevent them from causing trouble. This mechanism is called fencing or *STONITH*.



Important: No Support Without STONITH

A cluster without STONITH is not supported.

For a supported HA setup, ensure the following:

- Each node in the High Availability cluster needs to have at least one STONITH device (usually a hardware device). We strongly recommend multiple STONITH devices per node, unless STONITH Block Device (SBD) is used.
- The global cluster options `stonith-enabled` and `startup-fencing` must be set to `true`. These options are set automatically when deploying the `Pacemaker` barclamp. When you change them, you will lose support.
- When deploying the `Pacemaker` service, select a *STONITH: Configuration mode for STONITH* that matches your setup. If your STONITH devices support the IPMI protocol, choosing the IPMI option is the easiest way to configure STONITH. Another alternative is SBD. It provides a way to enable STONITH and fencing in clusters without external power switches, but it requires shared storage. For SBD requirements, see http://linux-ha.org/wiki/SBD_Fencing, section *Requirements*.

For more information, refer to the *High Availability Guide*, available at <https://documentation.suse.com/sle-ha/15-SP1/>. Especially read the following chapters: *Configuration and Administration Basics*, and *Fencing and STONITH*, *Storage Protection*.

Network Configuration



Important: Redundant Communication Paths

For a supported HA setup, it is required to set up cluster communication via two or more redundant paths. For this purpose, use network device bonding and team network mode in your Crowbar network setup. For details, see *Section 2.1.2.3, "Team Network Mode"*. At least two Ethernet cards per cluster node are required for network redundancy. We advise using team network mode everywhere (not only between the cluster nodes) to ensure redundancy.

For more information, refer to the *High Availability Guide*, available at <https://documentation.suse.com/sle-ha/15-SP1/>. Especially read the following chapter: *Network Device Bonding*.

Using a second communication channel (ring) in Corosync (as an alternative to network device bonding) is not supported yet in SUSE OpenStack Cloud Crowbar. By default, SUSE OpenStack Cloud Crowbar uses the admin network (typically `eth0`) for the first Corosync ring.

Important: Dedicated Networks

The `corosync` network communication layer is crucial to the health of the cluster. `corosync` traffic always goes over the admin network.

- Use redundant communication paths for the `corosync` network communication layer.
- Do not place the `corosync` network communication layer on interfaces shared with any other networks that could experience heavy load, such as the OpenStack public / private / SDN / storage networks.

Similarly, if SBD over iSCSI is used as a STONITH device (see *STONITH*), do not place the iSCSI traffic on interfaces that could experience heavy load, because this might disrupt the SBD mechanism.

Storage Requirements

When using SBD as STONITH device, additional requirements apply for the shared storage. For details, see http://linux-ha.org/wiki/SBD_Fencing, section *Requirements*.

2.6.6 For More Information

For a basic understanding and detailed information on the SUSE Linux Enterprise High Availability Extension (including the Pacemaker cluster stack), read the *High Availability Guide*. It is available at <https://documentation.suse.com/sle-ha/15-SP1/>.

In addition to the chapters mentioned in *Section 2.6.5, "Cluster Requirements and Recommendations"*, the following chapters are especially recommended:

- *Product Overview*
- *Configuration and Administration Basics*

The *High Availability Guide* also provides comprehensive information about the cluster management tools with which you can view and check the cluster status in SUSE OpenStack Cloud Crowbar. They can also be used to look up details like configuration of cluster resources or global cluster options. Read the following chapters for more information:

- HA Web Console: *Configuring and Managing Cluster Resources (Web Interface)*
- `crm.sh`: *Configuring and Managing Cluster Resources (Command Line)*

2.7 Summary: Considerations and Requirements

As outlined above, there are some important considerations to be made before deploying SUSE OpenStack Cloud. The following briefly summarizes what was discussed in detail in this chapter. Keep in mind that as of SUSE OpenStack Cloud Crowbar 8, it is not possible to change some aspects such as the network setup when SUSE OpenStack Cloud is deployed!

NETWORK

- If you do not want to stick with the default networks and addresses, define custom networks and addresses. You need five different networks. If you need to separate the admin and the BMC network, a sixth network is required. See [Section 2.1, "Network"](#) for details. Networks that share interfaces need to be configured as VLANs.
- The SUSE OpenStack Cloud networks are completely isolated, therefore it is not required to use public IP addresses for them. A class C network as used in this documentation may not provide enough addresses for a cloud that is supposed to grow. You may alternatively choose addresses from a class B or A network.
- Determine how to allocate addresses from your network. Make sure not to allocate IP addresses twice. See [Section 2.1.1, "Network Address Allocation"](#) for the default allocation scheme.
- Define which network mode to use. Keep in mind that all machines within the cloud (including the Administration Server) will be set up with the chosen mode and therefore need to meet the hardware requirements. See [Section 2.1.2, "Network Modes"](#) for details.
- Define how to access the admin and BMC network(s): no access from the outside (no action is required), via an external gateway (gateway needs to be provided), or via bastion network. See [Section 2.1.3, "Accessing the Administration Server via a Bastion Network"](#) for details.
- Provide a gateway to access the public network (public, nova-floating).

- Make sure the Administration Server's host name is correctly configured (`hostname -f` needs to return a fully qualified host name). If this is not the case, run `YaST > Network Services > Hostnames` and add a fully qualified host name.
- Prepare a list of MAC addresses and the intended use of the corresponding host for all OpenStack nodes.

UPDATE REPOSITORIES

- Depending on your network setup you have different options for providing up-to-date update repositories for SUSE Linux Enterprise Server and SUSE OpenStack Cloud for SUSE OpenStack Cloud deployment: using an existing SMT or SUSE Manager server, installing SMT on the Administration Server, synchronizing data with an existing repository, mounting remote repositories, or using physical media. Choose the option that best matches your needs.

STORAGE

- Decide whether you want to deploy the object storage service swift. If so, you need to deploy at least two nodes with sufficient disk space exclusively dedicated to swift.
- Decide which back-end to use with cinder. If using the `raw` back-end (local disks) we strongly recommend using a separate node equipped with several hard disks for deploying `cinder-volume`. Ceph needs a minimum of four exclusive nodes with sufficient disk space.
- Make sure all Compute Nodes are equipped with sufficient hard disk space.

SSL ENCRYPTION

- Decide whether to use different SSL certificates for the services and the API, or whether to use a single certificate.
- Get one or more SSL certificates certified by a trusted third party source.

HARDWARE AND SOFTWARE REQUIREMENTS

- Make sure the hardware requirements for the different node types are met.
- Make sure to have all required software at hand.

2.8 Overview of the SUSE OpenStack Cloud Installation

Deploying and installing SUSE OpenStack Cloud Crowbar is a multi-step process. Start by deploying a basic SUSE Linux Enterprise Server installation and the SUSE OpenStack Cloud Crowbar add-on product to the Administration Server. Then the product and update repositories need to be set up and the SUSE OpenStack Cloud network needs to be configured. Next, complete the Administration Server setup. After the Administration Server is ready, you can start deploying and configuring the OpenStack nodes. The complete node deployment is done automatically via Crowbar and Chef from the Administration Server. All you need to do is to boot the nodes using PXE and to deploy the OpenStack components to them.

1. Install SUSE Linux Enterprise Server 12 SP4 on the Administration Server with the add-on product SUSE OpenStack Cloud. Optionally select the Subscription Management Tool (SMT) pattern for installation. See *Chapter 3, Installing the Administration Server*.
2. Optionally set up and configure the SMT server on the Administration Server. See *Chapter 4, Installing and Setting Up an SMT Server on the Administration Server (Optional)*.
3. Make all required software repositories available on the Administration Server. See *Chapter 5, Software Repository Setup*.
4. Set up the network on the Administration Server. See *Chapter 6, Service Configuration: Administration Server Network Configuration*.
5. Perform the Crowbar setup to configure the SUSE OpenStack Cloud network and to make the repository locations known. When the configuration is done, start the SUSE OpenStack Cloud Crowbar installation. See *Chapter 7, Crowbar Setup*.
6. Boot all nodes onto which the OpenStack components should be deployed using PXE and allocate them in the Crowbar Web interface to start the automatic SUSE Linux Enterprise Server installation. See *Chapter 11, Installing the OpenStack Nodes*.
7. Configure and deploy the OpenStack components via the Crowbar Web interface or command line tools. See *Chapter 12, Deploying the OpenStack Services*.
8. When all OpenStack components are up and running, SUSE OpenStack Cloud is ready. The cloud administrator can now upload images to enable users to start deploying instances. See the *Administrator Guide* and the *Supplement to Administrator Guide and User Guide*.

II Setting Up the Administration Server

- 3 Installing the Administration Server **46**
- 4 Installing and Setting Up an SMT Server on the Administration Server (Optional) **49**
- 5 Software Repository Setup **53**
- 6 Service Configuration: Administration Server Network Configuration **64**
- 7 Crowbar Setup **66**
- 8 Starting the SUSE OpenStack Cloud Crowbar installation **97**
- 9 Customizing Crowbar **101**

3 Installing the Administration Server

In this chapter you will learn how to install the Administration Server from scratch. It will run on SUSE Linux Enterprise Server 12 SP4 and include the SUSE OpenStack Cloud Crowbar extension and, optionally, the Subscription Management Tool (SMT) server. Prior to starting the installation, refer to [Section 2.4, “Hardware Requirements”](#) and [Section 2.5, “Software Requirements”](#).

3.1 Starting the Operating System Installation

Start the installation by booting into the SUSE Linux Enterprise Server 12 SP4 installation system. For an overview of a default SUSE Linux Enterprise Server installation, refer to <https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#cha-install>.

The following sections will only cover the differences from the default installation process.

3.2 Registration and Online Updates

Registering SUSE Linux Enterprise Server 12 SP4 during the installation process is required for getting product updates and for installing the SUSE OpenStack Cloud Crowbar extension.

After a successful registration you will be asked whether to add the update repositories. If you agree, the latest updates will automatically be installed, ensuring that your system is on the latest patch level after the initial installation. We strongly recommend adding the update repositories immediately. If you choose to skip this step you need to perform an online update later, before starting the SUSE OpenStack Cloud Crowbar installation.



Note: SUSE Login Required

To register a product, you need to have a SUSE login. If you do not have such a login, create it at <http://www.suse.com/>.

3.3 Installing the SUSE OpenStack Cloud Crowbar Extension

SUSE OpenStack Cloud is an extension to SUSE Linux Enterprise Server. Installing it during the SUSE Linux Enterprise Server installation is the easiest and recommended way to set up the Administration Server. To get access to the extension selection dialog, you need to register SUSE Linux Enterprise Server 12 SP4 during the installation. After a successful registration, the SUSE Linux Enterprise Server 12 SP4 installation continues with the *Extension & Module Selection*. Choose *SUSE OpenStack Cloud Crowbar 9* and provide the registration key you obtained by purchasing SUSE OpenStack Cloud Crowbar. The registration and the extension installation require an Internet connection.

Alternatively, install the SUSE OpenStack Cloud Crowbar after the SUSE Linux Enterprise Server 12 SP4 installation via *YaST > Software > Add-On Products*. For details, refer to the section <https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#sec-yast-install-modules>.

3.4 Partitioning

Currently, Crowbar requires `/opt` to be writable. We recommend creating a separate partition or volume formatted with XFS for `/srv` with a size of at least 30 GB.

The default file system on SUSE Linux Enterprise Server 12 SP4 is Btrfs with snapshots enabled. SUSE OpenStack Cloud Crowbar installs into `/opt`, a directory that is excluded from snapshots. Reverting to a snapshot may therefore break the SUSE OpenStack Cloud Crowbar installation. We recommend disabling Btrfs snapshots on the Administration Server.

Help on using the partitioning tool is available at the section <https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#sec-expert-partitioner>.

3.5 Installation Settings

In the final installation step, *Installation Settings*, you need to adjust the software selection and the firewall settings for your Administration Server setup. For more information refer to the <https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#sec-yast-install-perform>.

3.5.1 Software Selection

Installing a minimal base system is sufficient to set up the Administration Server. The following patterns are the minimum required:

- *Base System*
- *Minimal System (Appliances)*
- *Meta Package for Pattern `cloud_admin`* (in case you have chosen to install the SUSE OpenStack Cloud Crowbar Extension)
- *Subscription Management Tool* (optional, also see [Tip: Installing a Local SMT Server \(Optional\)](#))



Tip: Installing a Local SMT Server (Optional)

If you do not have a SUSE Manager or SMT server in your organization, or are planning to manually update the repositories required for deployment of the SUSE OpenStack Cloud nodes, you need to set up an SMT server on the Administration Server. Choose the pattern *Subscription Management Tool* in addition to the patterns listed above to install the SMT server software.

3.5.2 Firewall Settings

SUSE OpenStack Cloud Crowbar requires disabling the firewall on the Administration Server. You can disable the firewall during installation in the *Firewall and SSH* section. If your environment requires a firewall to be active at this stage of the installation, you can disable the firewall during your final network configuration (see [Chapter 6, Service Configuration: Administration Server Network Configuration](#)). Optionally, you can also enable SSH access to the Administration Server in this section.



Warning: HTTP_PROXY and NO_PROXY

Setting `HTTP_PROXY` without properly configuring `NO_PROXY` for the Administration Server might result in `chef-client` failing in non-obvious ways.

4 Installing and Setting Up an SMT Server on the Administration Server (Optional)

One way to provide the repositories needed to set up the nodes in SUSE OpenStack Cloud is to install a Subscription Management Tool (SMT) server on the Administration Server, and then mirror all repositories from SUSE Customer Center via this server. Installing an SMT server on the Administration Server is optional. If your organization already provides an SMT server or a SUSE Manager server that can be accessed from the Administration Server, skip this step.



Important: Use of SMT Server and Ports

When installing an SMT server on the Administration Server, use it exclusively for SUSE OpenStack Cloud Crowbar. To use the SMT server for other products, run it outside of SUSE OpenStack Cloud Crowbar. Make sure it can be accessed from the Administration Server for mirroring the repositories needed for SUSE OpenStack Cloud Crowbar.

When the SMT server is installed on the Administration Server, Crowbar provides the mirrored repositories on port 8091.

4.1 SMT Installation

If you have not installed the SMT server during the initial Administration Server installation as suggested in *Section 3.5.1, "Software Selection"*, run the following command to install it:

```
sudo zypper in -t pattern smt
```

4.2 SMT Configuration

No matter whether the SMT server was installed during the initial installation or in the running system, it needs to be configured with the following steps.



Note: Prerequisites

To configure the SMT server, a SUSE account is required. If you do not have such an account, register at <http://www.suse.com/>. All products and extensions for which you want to mirror updates with the SMT server should be registered at the SUSE Customer Center (<http://scc.suse.com/>).

1. Configuring the SMT server requires you to have your mirroring credentials (user name and password) and your registration e-mail address at hand. To access them, proceed as follows:
 - a. Open a Web browser and log in to the SUSE Customer Center at <http://scc.suse.com/>.
 - b. Click your name to see the e-mail address which you have registered.
 - c. Click *Organization > Organization Credentials* to obtain your mirroring credentials (user name and password).
2. Start *YaST > Network Services > SMT Configuration Wizard*.
3. Activate *Enable Subscription Management Tool Service (SMT)*.
4. Enter the *Customer Center Configuration* data as follows:

Use Custom Server: Do not activate this option
User: The user name you retrieved from the SUSE Customer Center
Password: The password you retrieved from the SUSE Customer Center
Check your input with *Test*. If the test does not return success, check the credentials you entered.
5. Enter the e-mail address you retrieved from the SUSE Customer Center at *SCC E-Mail Used for Registration*.
6. *Your SMT Server URL* shows the HTTP address of your server. Usually it should not be necessary to change it.
7. Select *Next* to proceed to step two of the *SMT Configuration Wizard*.
8. Enter a *Database Password for SMT User* and confirm it by entering it once again.
9. Enter one or more e-mail addresses to which SMT status reports are sent by selecting *Add*.

10. Select *Next* to save your SMT configuration. When setting up the database you will be prompted for the MariaDB root password. If you have not already created one then create it in this step. Note that this is the global MariaDB root password, not the database password for the SMT user you specified before.

The SMT server requires a server certificate at `/etc/pki/trust/anchors/YaST-CA.pem`. Choose *Run CA Management*, provide a password and choose *Next* to create such a certificate. If your organization already provides a CA certificate, *Skip* this step and import the certificate via *YaST > Security and Users > CA Management* after the SMT configuration is done. See <https://documentation.suse.com/sles/15-SP1/single-html/SLES-security/#cha-security-yast-security> for more information.

After you complete your configuration a synchronization check with the SUSE Customer Center will run, which may take several minutes.

4.3 Setting up Repository Mirroring on the SMT Server

The final step in setting up the SMT server is configuring it to mirror the repositories needed for SUSE OpenStack Cloud. The SMT server mirrors the repositories from the SUSE Customer Center. Make sure to have the appropriate subscriptions registered in SUSE Customer Center with the same e-mail address you specified when configuring SMT. For details on the required subscriptions refer to [Section 2.5, "Software Requirements"](#).

4.3.1 Adding Mandatory Repositories

Mirroring the SUSE Linux Enterprise Server 12 SP4 and SUSE OpenStack Cloud Crowbar 9 repositories is mandatory. Run the following commands as user `root` to add them to the list of mirrored repositories:

```
for REPO in SLES12-SP4-{Pool,Updates} SUSE-OpenStack-Cloud-Crowbar-9-{Pool,Updates}; do
  smt-repos $REPO sle-12-x86_64 -e
done
```

4.3.2 Adding Optional Repositories

The following optional repositories provide high availability and storage:

High Availability

For the optional HA setup you need to mirror the SLE-HA12-SP4 repositories. Run the following commands as user `root` to add them to the list of mirrored repositories:

```
for REPO in SLE-HA12-SP4-{Pool,Updates}; do
    smt-repos $REPO sle-12-x86_64 -e
done
```

SUSE Enterprise Storage

The SUSE Enterprise Storage repositories are needed if you plan to use an external Ceph with SUSE OpenStack Cloud. Run the following commands as user `root` to add them to the list of mirrored repositories:

```
for REPO in SUSE-Enterprise-Storage-5-{Pool,Updates}; do
    smt-repos $REPO sle-12-x86_64 -e
done
```

4.3.3 Updating the Repositories

New repositories added to SMT must be updated immediately by running the following command as user `root`:

```
smt-mirror -L /var/log/smt/smt-mirror.log
```

This command will download several GB of patches. This process may last up to several hours. A log file is written to `/var/log/smt/smt-mirror.log`. After this first manual update the repositories are updated automatically via cron job. A list of all repositories and their location in the file system on the Administration Server can be found at [Table 5.2, "SMT Repositories Hosted on the Administration Server"](#).

4.4 For More Information

For detailed information about SMT refer to the Subscription Management Tool manual at <https://documentation.suse.com/sles/12-SP5/single-html/SLES-smt/>.

5 Software Repository Setup

Nodes in SUSE OpenStack Cloud are automatically installed from the Administration Server. For this to happen, software repositories containing products, extensions, and the respective updates for all software need to be available on or accessible from the Administration Server. In this configuration step, these repositories are made available. There are two types of repositories:

Product Media Repositories: Product media repositories are copies of the installation media. They need to be directly copied to the Administration Server, “loop-mounted” from an iso image, or mounted from a remote server via NFS. Affected are SUSE Linux Enterprise Server 12 SP4 and SUSE OpenStack Cloud Crowbar 9. These are static repositories; they do not change or receive updates. See [Section 5.1, “Copying the Product Media Repositories”](#) for setup instructions.

Update and Pool Repositories: Update and Pool repositories are provided by the SUSE Customer Center. They contain all updates and patches for the products and extensions. To make them available for SUSE OpenStack Cloud they need to be mirrored from the SUSE Customer Center. Since their content is regularly updated, they must be kept in synchronization with SUSE Customer Center. For these purposes, SUSE provides either the Subscription Management Tool (SMT) or the SUSE Manager.

5.1 Copying the Product Media Repositories

The files in the product repositories for SUSE Linux Enterprise Server and SUSE OpenStack Cloud do not change, therefore they do not need to be synchronized with a remote source. It is sufficient to either copy the data (from a remote host or the installation media), to mount the product repository from a remote server via NFS, or to loop mount a copy of the installation images.

Important: No Symbolic Links for the SUSE Linux Enterprise Server Repository

Note that the SUSE Linux Enterprise Server product repository *must* be directly available from the local directory listed below. It is not possible to use a symbolic link to a directory located elsewhere, since this will cause booting via PXE to fail.

Tip: Providing the SUSE OpenStack Cloud Crowbar Repository via HTTP

The SUSE Linux Enterprise Server product repositories need to be available locally to enable booting via PXE for node deployment. The SUSE OpenStack Cloud Crowbar repository may also be served via HTTP from a remote host. In this case, enter the URL to the [Cloud](#) repository as described in [Section 7.4, "Repositories"](#).

We recommend copying the data to the Administration Server as the best solution. It does not require much hard disk space (approximately 900 MB). Nor does it require the Administration Server to access a remote host from a different network.

The following product media must be copied to the specified directories:

TABLE 5.1: LOCAL PRODUCT REPOSITORIES FOR SUSE OPENSTACK CLOUD

Repository	Directory
SUSE Linux Enterprise Server 12 SP4 DVD #1	<u>/srv/tftpboot/suse-12.4/x86_64/install</u>
SUSE OpenStack Cloud Crowbar 9 DVD #1	<u>/srv/tftpboot/suse-12.4/x86_64/repos/Cloud</u>

The data can be copied by a variety of methods:

Copying from the Installation Media

We recommend using **rsync** for copying. If the installation data is located on a removable device, make sure to mount it first (for example, after inserting the DVD1 in the Administration Server and waiting for the device to become ready):

SUSE Linux Enterprise Server 12 SP4 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/install
mount /dev/dvd /mnt
rsync -avP /mnt/ /srv/tftpboot/suse-12.4/x86_64/install/
umount /mnt
```

SUSE OpenStack Cloud Crowbar 9 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/repos/Cloud
mount /dev/dvd /mnt
rsync -avP /mnt/ /srv/tftpboot/suse-12.4/x86_64/repos/Cloud/
umount /mnt
```

Copying from a Remote Host

If the data is provided by a remote machine, log in to that machine and push the data to the Administration Server (which has the IP address 192.168.124.10 in the following example):

SUSE Linux Enterprise Server 12 SP4 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/install
rsync -avPz /data/SLES-12-SP4/DVD1/ 192.168.124.10:/srv/tftpboot/suse-12.4/x86_64/
install/
```

SUSE OpenStack Cloud Crowbar 9 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/repos/Cloud
rsync -avPz /data/SUSE-OPENSTACK-CLOUD//DVD1/ 192.168.124.10:/srv/tftpboot/
suse-12.4/x86_64/repos/Cloud/
```

Mounting from an NFS Server

If the installation data is provided via NFS by a remote machine, mount the respective shares as follows. To automatically mount these directories either create entries in /etc/fstab or set up the automounter.

SUSE Linux Enterprise Server 12 SP4 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/install
mount -t nfs nfs.example.com:/exports/SLES-12-SP4/x86_64/DVD1/ /srv/tftpboot/
suse-12.4/x86_64/install
```

SUSE OpenStack Cloud Crowbar 9 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/repos/Cloud/
mount -t nfs nfs.example.com:/exports/SUSE-OPENSTACK-CLOUD/DVD1/ /srv/tftpboot/
suse-12.4/x86_64/repos/Cloud
```

Mounting the ISO Images

The product repositories can also be made available by copying the respective ISO images to the Administration Server and mounting them. To automatically mount these directories either create entries in /etc/fstab or set up the automounter.

SUSE Linux Enterprise Server 12 SP4 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/install/
mount -o loop /local/SLES-12-SP4-x86_64-DVD1.iso /srv/tftpboot/suse-12.4/x86_64/
install
```

SUSE OpenStack Cloud Crowbar 9 DVD#1

```
mkdir -p /srv/tftpboot/suse-12.4/x86_64/repos/Cloud/
mount -o loop /local/SUSE-OPENSTACK-CLOUD-9-x86_64-DVD1.iso /srv/tftpboot/suse-12.4/
x86_64/repos/Cloud
```


5.2 Update and Pool Repositories

Update and Pool Repositories are required on the Administration Server to set up and maintain the SUSE OpenStack Cloud nodes. They are provided by SUSE Customer Center and contain all software packages needed to install SUSE Linux Enterprise Server 12 SP4 and the extensions (pool repositories). In addition, they contain all updates and patches (update repositories). Update repositories are used when deploying the nodes that build SUSE OpenStack Cloud to ensure they are initially equipped with the latest software versions available.

The repositories can be made available on the Administration Server using one or more of the following methods:

- [Section 5.2.1, “Repositories Hosted on an SMT Server Installed on the Administration Server”](#)
- [Section 5.2.2, “Repositories Hosted on a Remote SMT Server”](#)
- [Section 5.2.3, “Repositories Hosted on a SUSE Manager Server”](#)
- [Section 5.2.4, “Alternative Ways to Make the Repositories Available”](#)

5.2.1 Repositories Hosted on an SMT Server Installed on the Administration Server

When all update and pool repositories are managed by an SMT server installed on the Administration Server (see [Chapter 4, Installing and Setting Up an SMT Server on the Administration Server \(Optional\)](#)), make sure the repository location in YaST Crowbar is set to *Local SMT Server* (this is the default). For details, see [Section 7.4, “Repositories”](#). No further action is required. The SUSE OpenStack Cloud Crowbar installation automatically detects all available repositories.

5.2.2 Repositories Hosted on a Remote SMT Server

To use repositories from a remote SMT server, you first need to make sure all required repositories are mirrored on the server. Refer to [Section 4.3, “Setting up Repository Mirroring on the SMT Server”](#) for more information. When all update and pool repositories are managed by a remote SMT server, make sure the repository location in YaST Crowbar is set to *Remote SMT Server*. For details, see [Section 7.4, “Repositories”](#). No further action is required. The SUSE OpenStack Cloud Crowbar installation automatically detects all available repositories.



Note: Accessing an External SMT Server

When using an external SMT server, it needs to be reachable by all nodes. This means that the SMT server either needs to be part of the admin network or it needs to be accessible via the default route of the nodes. The latter can be either the gateway of the admin network or the gateway of the public network.

5.2.3 Repositories Hosted on a SUSE Manager Server

To use repositories from SUSE Manager you first need to make sure all required products and extensions are registered, and the corresponding channels are mirrored in SUSE Manager (refer to [Table 5.4, “SUSE Manager Repositories \(Channels\)”](#) for a list of channels).



Important: Accessing a SUSE Manager Server

An external SUSE Manager server needs to be accessible to *all* nodes in SUSE OpenStack Cloud. The network hosting the SUSE Manager server must be added to the network definitions as described in [Section 7.5.8, “Providing Access to External Networks”](#).

By default SUSE Manager does not expose repositories for direct access. To access them via HTTPS, you need to create a *Distribution* for auto-installation for the SUSE Linux Enterprise Server 12 SP4 (x86_64) product. Creating this distribution makes the update repositories for this product available, including the repositories for all registered add-on products (like SUSE OpenStack Cloud Crowbar, SLES High Availability Extension and SUSE Enterprise Storage). Instructions for creating a distribution are in the SUSE Manager documentation in <https://documentation.suse.com/suma/4.0/>.

During the distribution setups you need to provide a *Label* for each the distribution. This label will be part of the URL under which the repositories are available. We recommend choosing a name consisting of characters that do not need to be URL-encoded. In [Table 5.4, “SUSE Manager Repositories \(Channels\)”](#) we assume the following label has been provided: `sles12-sp4-x86_64`. When all update and pool repositories are managed by a SUSE Manager server, make sure the repository location in YaST Crowbar is set to *SUSE Manager Server*. For details, see [Section 7.4, “Repositories”](#). No further action is required. The SUSE OpenStack Cloud Crowbar installation automatically detects all available repositories.

The autoinstallation tree provided by SUSE Manager does not provide the SLES Pool repository. Although this repository is not used for node installation, it needs to be present. To work around this issue, it is sufficient to create an empty Pool repository for SUSE Linux Enterprise Server 12 SP4:

```
mkdir /srv/tftpboot/suse-12.4/x86_64/repos/SLES12-SP4-Pool/  
createrepo /srv/tftpboot/suse-12.4/x86_64/repos/SLES12-SP4-Pool/
```

5.2.4 Alternative Ways to Make the Repositories Available

If you want to keep your SUSE OpenStack Cloud network as isolated from the company network as possible, or your infrastructure does not allow accessing a SUSE Manager or an SMT server, you can alternatively provide access to the required repositories by one of the following methods:

- Mount the repositories from a remote server.
- Synchronize the repositories from a remote server (for example via [rsync](#) and cron).
- Manually synchronize the update repositories from removable media.

We strongly recommend making the repositories available at the default locations on the Administration Server as listed in [Table 5.5, “Default Repository Locations on the Administration Server”](#). When choosing these locations, it is sufficient to set the repository location in YaST Crowbar to *Custom*. You do not need to specify a detailed location for each repository. Refer to [Section 7.4, “Repositories”](#) for details. If you prefer to use different locations, you need to announce each location with YaST Crowbar.

5.3 Software Repository Sources for the Administration Server Operating System

During the installation of the Administration Server, repository locations for SUSE Linux Enterprise Server 12 SP4 are automatically added to the Administration Server. They point to the source used to install the Administration Server and to the SUSE Customer Center. These repository locations have no influence on the repositories used to set up nodes in the cloud. They are solely used to maintain and update the Administration Server itself.

However, as the Administration Server and all nodes in the cloud use the same operating system—SUSE Linux Enterprise Server 12 SP4—it makes sense to use the same repositories for the cloud and the Administration Server. To avoid downloading the same patches twice, change this setup so that the repositories set up for SUSE OpenStack Cloud deployment are also used on the Administration Server.

To do so, you need to disable or delete all services. In a second step all SUSE Linux Enterprise Server and SUSE OpenStack Cloud repositories need to be edited to point to the alternative sources. Use either Zypper or YaST to edit the repository setup. Note that changing the repository setup on the Administration Server is optional.

5.4 Repository Locations

The following tables show the locations of all repositories that can be used for SUSE OpenStack Cloud Crowbar.

TABLE 5.2: SMT REPOSITORIES HOSTED ON THE ADMINISTRATION SERVER

Repository	Directory
Mandatory Repositories	
SLES12-SP4-Pool	<u>/srv/www/htdocs/repo/SUSE/Products/SLE-SERVER/12-SP4/x86_64/product/</u>
SLES12-SP4-Updates	<u>/srv/www/htdocs/repo/SUSE/Updates/SLE-SERVER/12-SP4/x86_64/update/</u>
SUSE-OpenStack-Cloud-Crowbar-9-Pool	<u>/srv/www/htdocs/repo/SUSE/Products/OpenStack-Cloud-Crowbar/9/x86_64/product/</u>
SUSE-OpenStack-Cloud-Crowbar-9-Updates	<u>/srv/www/htdocs/repo/SUSE/Updates/OpenStack-Cloud-Crowbar/9/x86_64/update/</u>
Optional Repositories	

Repository	Directory
SLE-HA12-SP4-Pool	/srv/www/htdocs/repo/SUSE/Products/SLE-HA/12-SP4/x86_64/product/
SLE-HA12-SP4-Updates	/srv/www/htdocs/repo/SUSE/Updates/SLE-HA/12-SP4/x86_64/update/
SUSE-Enterprise-Storage-5-Pool	/srv/www/htdocs/repo/SUSE/Products/Storage/5/x86_64/product/
SUSE-Enterprise-Storage-5-Updates	/srv/www/htdocs/repo/SUSE/Updates/Storage/5/x86_64/update/

TABLE 5.3: SMT REPOSITORIES HOSTED ON A REMOTE SERVER

Repository	URI
Mandatory Repositories	
SLES12-SP4-Pool	http://smt.example.com/repo/SUSE/Products/SLE-SERVER/12-SP4/x86_64/product/
SLES12-SP4-Updates	http://smt.example.com/repo/SUSE/Updates/SLE-SERVER/12-SP4/x86_64/update/
SUSE-OpenStack-Cloud-Crowbar-9-Pool	http://smt.example.com/repo/SUSE/Products/OpenStack-Cloud/9/x86_64/product/
SUSE-OpenStack-Cloud-Crowbar-9-Updates	http://smt.example.com/repo/SUSE/Updates/OpenStack-Cloud/9/x86_64/update/
Optional Repositories	
SLE-HA12-SP4-Pool	http://smt.example.com/repo/SUSE/Products/SLE-HA/12-SP4/x86_64/product/

Repository	URI
SLE-HA12-SP4-Up- dates	http://smt.example.com/repo/SUSE/Updates/SLE-HA/12-SP4/x86_64/update/
SUSE-Enterprise-Stor- age-5-Pool	http://smt.example.com/repo/SUSE/Products/Storage/4/x86_64/product/
SUSE-Enterprise-Stor- age-5-Updates	http://smt.example.com/repo/SUSE/Updates/Storage/4/x86_64/update/

TABLE 5.4: SUSE MANAGER REPOSITORIES (CHANNELS)

Repository	URL
Mandatory Repositories	
SLES12-SP4-Updates	http://manager.example.com/ks/dist/child/sles12-sp4-updates-x86_64/sles12-sp4-x86_64/
SUSE-OpenS- tack-Cloud-Crow- bar-9-Pool	http://manager.example.com/ks/dist/child/suse-openstack-cloud-9-pool-x86_64/sles12-sp4-x86_64/
SUSE-OpenS- tack-Cloud-Crow- bar-9--Updates	http://manager.example.com/ks/dist/child/suse-openstack-cloud-9-updates-x86_64/sles12-sp4-x86_64/
Optional Repositories	
SLE-HA12-SP4-Pool	http://manager.example.com/ks/dist/child/sle-ha12-sp4-pool-x86_64/sles12-sp4-x86_64/
SLE-HA12-SP4-Up- dates	http://manager.example.com/ks/dist/child/sle-ha12-sp4-updates-x86_64/sles12-sp4-x86_64/
SUSE-Enterprise-Stor- age-5-Pool	http://manager.example.com/ks/dist/child/suse-enterprise-storage-2.1-pool-x86_64/sles12-sp4-x86_64/

Repository	URL
SUSE-Enterprise-Storage-5-Updates	http://manager.example.com/ks/dist/child/suse-enterprise-storage-5-updates-x86_64/sles12-sp4-x86_64/

The following table shows the recommended default repository locations to use when manually copying, synchronizing, or mounting the repositories. When choosing these locations, it is sufficient to set the repository location in YaST Crowbar to *Custom*. You do not need to specify a detailed location for each repository. Refer to [Section 5.2.4, “Alternative Ways to Make the Repositories Available”](#) and [Section 7.4, “Repositories”](#) for details.

TABLE 5.5: DEFAULT REPOSITORY LOCATIONS ON THE ADMINISTRATION SERVER

Channel	Directory on the Administration Server
Mandatory Repositories	
SLES12-SP4-Pool	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SLES12-SP4-Pool/</u>
SLES12-SP4-Updates	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SLES12-SP4-Updates/</u>
SUSE-OpenStack-Cloud-Crowbar-9-Pool	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SUSE-OpenStack-Cloud-Crowbar-9-Pool/</u>
SUSE-OpenStack-Cloud-Crowbar-9-Updates	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SUSE-OpenStack-Cloud-Crowbar-9-Updates</u>
Optional Repositories	
SLE-HA12-SP4-Pool	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SLE-HA12-SP4-Pool</u>
SLE-HA12-SP4-Updates	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SLE-HA12-SP4-Updates</u>
SUSE-Enterprise-Storage-5-Pool	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SUSE-Enterprise-Storage-5-Pool</u>
SUSE-Enterprise-Storage-5-Updates	<u>/srv/tftpboot/suse-12.4/x86_64/repos/SUSE-Enterprise-Storage-5-Updates</u>

6 Service Configuration: Administration Server Network Configuration

Prior to starting the SUSE OpenStack Cloud Crowbar installation, make sure the first network interface (`eth0`) gets a fixed IP address from the admin network. A host and domain name also need to be provided. Other interfaces will be automatically configured during the SUSE OpenStack Cloud Crowbar installation.

To configure the network interface proceed as follows:

1. Start *YaST* > *System* > *Network Settings*.
2. Switch to the *Overview* tab, select the interface with the *Device* identifier, `eth0` and choose *Edit*.
3. Switch to the *Address* tab and activate *Statically Assigned IP Address*. Provide an IPv4 *IP Address*, a *Subnet Mask*, and a fully qualified *Hostname*. Examples in this book assume the default IP address of `192.168.124.10` and a network mask of `255.255.255.0`. Using a different IP address requires adjusting the Crowbar configuration in a later step as described in *Chapter 7, Crowbar Setup*.
4. Check the settings on the *General* tab. The device needs to be activated *At Boot Time*. Confirm your settings with *Next*.
5. Back on the *Network Settings* dialog, switch to the *Routing* tab and enter a *Default IPv4 Gateway*. The address depends on whether you have provided an external gateway for the admin network. In that case, use the address of that gateway. If not, use `xxx.xxx.xxx.1`, for example, `192.168.124.1`. Confirm your settings with *OK*.
6. Choose *Hostname/DNS* from the *Network Settings* dialog and set the *Hostname* and *Domain Name*. Examples in this book assume `admin.cloud.example.com` for the host/domain name.
If the Administration Server has access to the outside, you can add additional name servers here that will automatically be used to forward requests. The Administration Server's name server will automatically be configured during the SUSE OpenStack Cloud Crowbar installation to forward requests for non-local records to those server(s).
7. Last, check if the firewall is disabled. Return to *YaST*'s main menu (*YaST Control Center*) and start *Security and Users* > *Firewall*. On *Start-Up* > *Service Start*, the firewall needs to be disabled. Confirm your settings with *Next*.



Important: Administration Server Domain Name and Host name

Setting up the SUSE OpenStack Cloud will also install a DNS server for all nodes in the cloud. The domain name you specify for the Administration Server will be used for the DNS zone. It is required to use a sub-domain such as `cloud.example.com`. See [Section 2.1.4, “DNS and Host Names”](#) for more information.

The host name and the FQDN need to be resolvable with `hostname -f`. Double-check whether `/etc/hosts` contains an appropriate entry for the Administration Server. It should look like the following:

```
192.168.124.10 admin.cloud.example.com admin
```

It is *not* possible to change the Administration Server host name or the FQDN after the SUSE OpenStack Cloud Crowbar installation has been completed.

7 Crowbar Setup

The YaST Crowbar module enables you to configure all networks within the cloud, to set up additional repositories, and to manage the Crowbar users. This module should be launched before starting the SUSE OpenStack Cloud Crowbar installation. To start this module, either run `yast crowbar` or `YaST > Miscellaneous > Crowbar`.

7.1 User Settings

In this section, you can manage the administration user for the Crowbar Web interface. Use the entries to change the user name and the password. The preconfigured user is `crowbar` (password `crowbar`). This administration user configured here has no relation to any existing system user on the Administration Server.

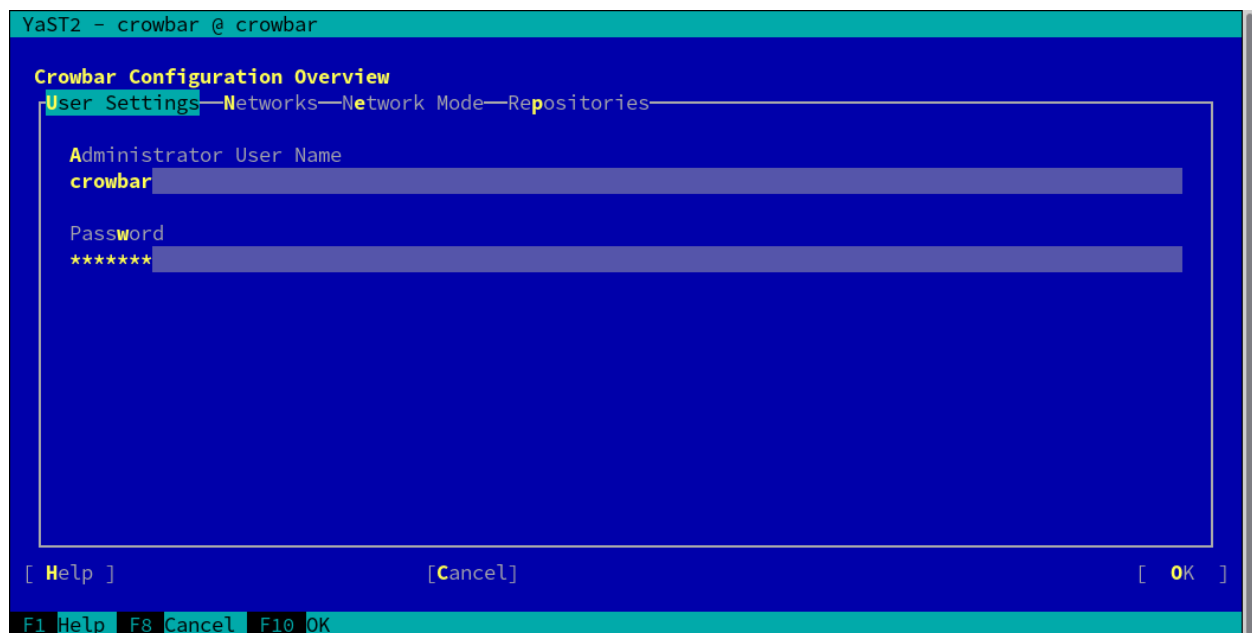


FIGURE 7.1: YAST CROWBAR SETUP: USER SETTINGS

7.2 Networks

Use the *Networks* tab to change the default network setup (described in [Section 2.1, “Network”](#)). Change the IP address assignment for each network under *Edit Ranges*. You may also add a bridge (*Add Bridge*) or a VLAN (*Use VLAN, VLAN ID*) to a network. Only change the latter two settings if you really know what you require; we recommend sticking with the defaults.

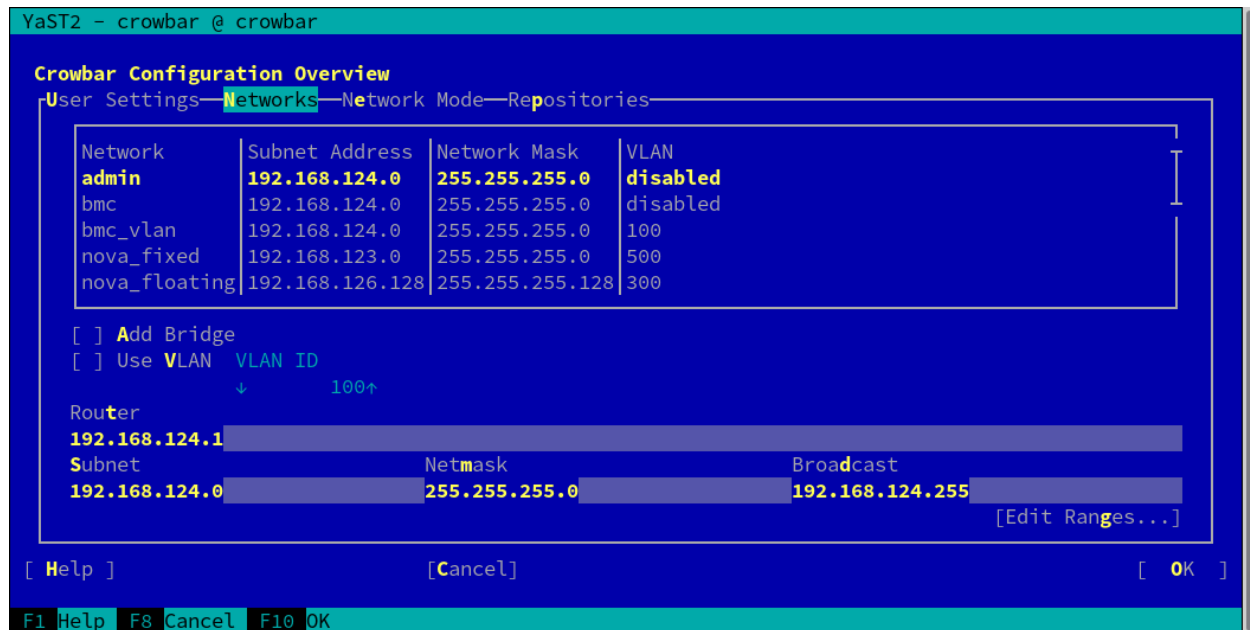


FIGURE 7.2: YAST CROWBAR SETUP: NETWORK SETTINGS

Warning: No Network Changes After Completing the SUSE OpenStack Cloud Crowbar installation

After you have completed the SUSE OpenStack Cloud Crowbar installation, you cannot change the network setup. If you do need to change it, you must completely set up the Administration Server again.

Important: VLAN Settings

As of SUSE OpenStack Cloud Crowbar 8, using a VLAN for the admin network is only supported on a native/untagged VLAN. If you need VLAN support for the admin network, it must be handled at switch level.

When changing the network configuration with YaST or by editing `/etc/crowbar/network.json`, you can define VLAN settings for each network. For the networks `nova-fixed` and `nova-floating`, however, special rules apply:

nova-fixed: The `USE VLAN` setting will be ignored. However, VLANs will automatically be used if deploying neutron with VLAN support (using the drivers `linuxbridge`, `openvswitch` plus `VLAN`, or `cisco_nexus`). In this case, you need to specify a correct `VLAN ID` for this network.

nova-floating: When using a VLAN for `nova-floating` (which is the default), the `USE VLAN` and `VLAN ID` settings for `nova-floating` and `public` default to the same.

You have the option of separating public and floating networks with a custom configuration. Configure your own separate floating network (not as a subnet of the public network), and give the floating network its own router. For example, define `nova-floating` as part of an external network with a custom `bridge-name`. When you are using different networks and OpenVSwitch is configured, the pre-defined `bridge-name` won't work.

Other, more flexible network mode setups, can be configured by manually editing the Crowbar network configuration files. See [Section 7.5, "Custom Network Configuration"](#) for more information. SUSE or a partner can assist you in creating a custom setup within the scope of a consulting services agreement. See <http://www.suse.com/consulting/> for more information on SUSE consulting.

7.2.1 Separating the Admin and the BMC Network

If you want to separate the admin and the BMC network, you must change the settings for the networks `bmc` and `bmc_vlan`. The `bmc_vlan` is used to generate a VLAN tagged interface on the Administration Server that can access the `bmc` network. The `bmc_vlan` needs to be in the same ranges as `bmc`, and `bmc` needs to have `VLAN` enabled.

TABLE 7.1: SEPARATE BMC NETWORK EXAMPLE CONFIGURATION

	<code>bmc</code>	<code>bmc_vlan</code>
Subnet	<code>192.168.128.0</code>	
Netmask	<code>255.255.255.0</code>	

	bmc	bmc_vlan
Router	192.168.128.1	
Broadcast	192.168.128.255	
Host Range	192.168.128.10 - 192.168.128.100	192.168.128.101 - 192.168.128.101
VLAN	yes	
VLAN ID	100	
Bridge	no	

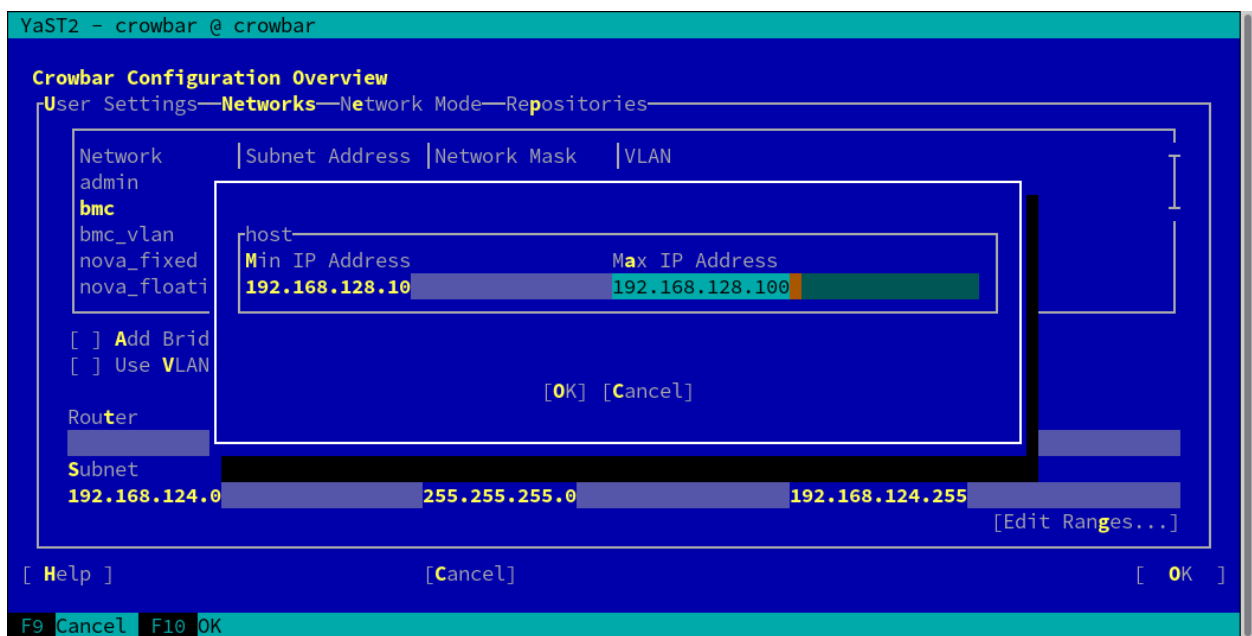


FIGURE 7.3: YAST CROWBAR SETUP: NETWORK SETTINGS FOR THE BMC NETWORK

7.3 Network Mode

On the *Network Mode* tab you can choose between *single*, *dual*, and *team*. In single mode, all traffic is handled by a single Ethernet card. Dual mode requires two Ethernet cards and separates traffic for private and public networks. See [Section 2.1.2, "Network Modes"](#) for details.

Team mode is similar to single mode, except that you combine several Ethernet cards to a “bond”. It is required for an HA setup of SUSE OpenStack Cloud. When choosing this mode, you also need to specify a *Bonding Policy*. This option lets you define whether to focus on reliability (fault tolerance), performance (load balancing), or a combination of both. You can choose from the following modes:

0 (balance-rr)

Default mode in SUSE OpenStack Cloud Crowbar. Packets are transmitted in round-robin fashion from the first to the last available interface. Provides fault tolerance and load balancing.

1 (active-backup)

Only one network interface is active. If it fails, a different interface becomes active. This setting is the default for SUSE OpenStack Cloud. Provides fault tolerance.

2 (balance-xor)

Traffic is split between all available interfaces based on the following policy: $[(\text{source MAC address XOR'd with destination MAC address XOR packet type ID}) \bmod \text{slave count}]$ Requires support from the switch. Provides fault tolerance and load balancing.

3 (broadcast)

All traffic is broadcast on all interfaces. Requires support from the switch. Provides fault tolerance.

4 (802.3ad)

Aggregates interfaces into groups that share the same speed and duplex settings. Requires **ethtool** support in the interface drivers, and a switch that supports and is configured for IEEE 802.3ad Dynamic link aggregation. Provides fault tolerance and load balancing.

5 (balance-tlb)

Adaptive transmit load balancing. Requires **ethtool** support in the interface drivers but no switch support. Provides fault tolerance and load balancing.

6 (balance-alb)

Adaptive load balancing. Requires **ethtool** support in the interface drivers but no switch support. Provides fault tolerance and load balancing.

For a more detailed description of the modes, see <https://www.kernel.org/doc/Documentation/networking/bonding.txt>.

7.3.1 Setting Up a Bastion Network

The *Network Mode* tab of the YaST Crowbar module also lets you set up a Bastion network. As outlined in [Section 2.1, “Network”](#), one way to access the Administration Server from a defined external network is via a Bastion network and a second network card (as opposed to providing an external gateway).

To set up the Bastion network, you need to have a static IP address for the Administration Server from the external network. The example configuration used below assumes that the external network from which to access the admin network has the following addresses. Adjust them according to your needs.

TABLE 7.2: EXAMPLE ADDRESSES FOR A BASTION NETWORK

Subnet	<u>10.10.1.0</u>
Netmask	<u>255.255.255.0</u>
Broadcast	<u>10.10.1.255</u>
Gateway	<u>10.10.1.1</u>
Static Administration Server address	<u>10.10.1.125</u>

In addition to the values above, you need to enter the *Physical Interface Mapping*. With this value you specify the Ethernet card that is used for the bastion network. See [Section 7.5.5, “Network Conduits”](#) for details on the syntax. The default value ?1g2 matches the second interface (“eth1”) of the system.

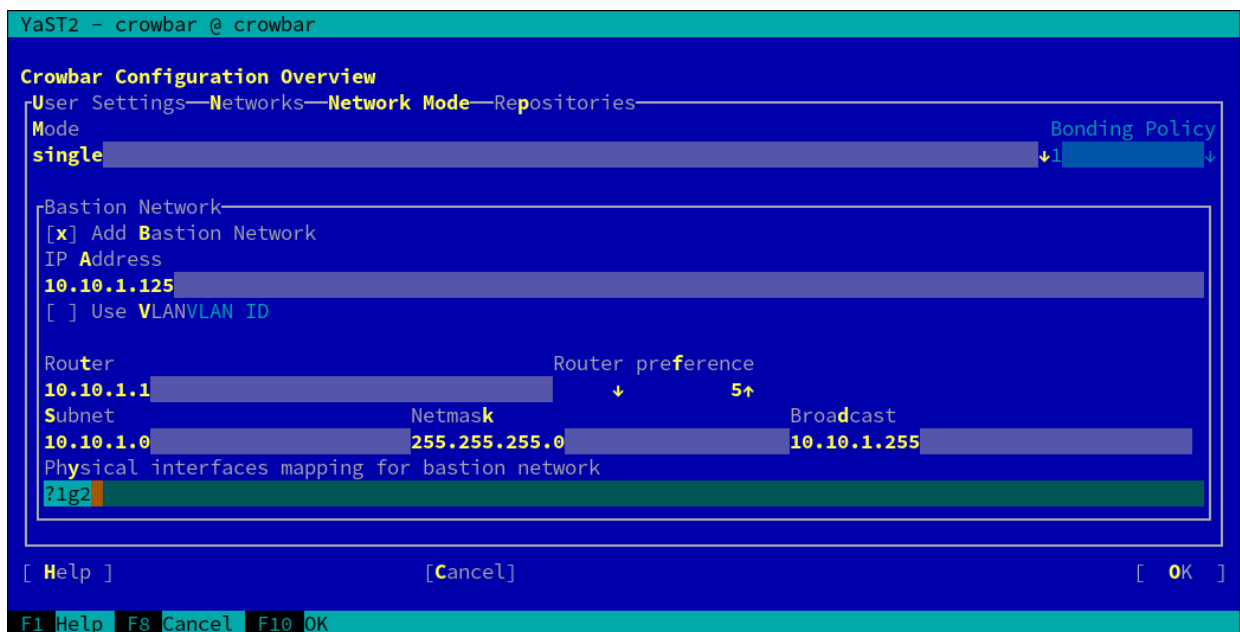


FIGURE 7.4: YAST CROWBAR SETUP: NETWORK SETTINGS FOR THE BASTION NETWORK



Warning: No Network Changes After Completing the SUSE OpenStack Cloud Crowbar installation

After you have completed the SUSE OpenStack Cloud Crowbar installation, you cannot change the network setup. If you do need to change it, you must completely set up the Administration Server again.



Important: Accessing Nodes From Outside the Bastion Network

The example configuration from above allows access to the SUSE OpenStack Cloud nodes from *within* the bastion network. If you want to access nodes from outside the bastion network, make the router for the bastion network the default router for the Administration Server. This is achieved by setting the value for the bastion network's *Router preference* entry to a lower value than the corresponding entry for the admin network. By default no router preference is set for the Administration Server—in this case, set the preference for the bastion network to 5.

If you use a Linux gateway between the outside and the bastion network, you also need to disable route verification (`rp_filter`) on the Administration Server. Do so by running the following command on the Administration Server:

```
echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
```


That command disables route verification for the current session, so the setting will not survive a reboot. Make it permanent by editing `/etc/sysctl.conf` and setting the value for `net.ipv4.conf.all.rp_filter` to `0`.

7.4 Repositories

This dialog lets you announce the locations of the product, pool, and update repositories (see [Chapter 5, Software Repository Setup](#) for details). You can choose between four alternatives:

Local SMT Server

If you have an SMT server installed on the Administration Server as explained in [Chapter 4, Installing and Setting Up an SMT Server on the Administration Server \(Optional\)](#), choose this option. The repository details do not need to be provided as they will be configured automatically. This option will be applied by default if the repository configuration has not been changed manually.

Remote SMT Server

If you use a remote SMT for *all* repositories, choose this option and provide the *Server URL* (in the form of `http://smt.example.com`). The repository details do not need to be provided, they will be configured automatically.

SUSE Manager Server

If you use a remote SUSE Manager server for *all* repositories, choose this option and provide the *Server URL* (in the form of `http://manager.example.com`).

Custom

If you use different sources for your repositories or are using non-standard locations, choose this option and manually provide a location for each repository. This can either be a local directory (`/srv/tftpboot/suse-12.4/x86_64/repos/SLES12-SP4-Pool/`) or a remote location (`http://manager.example.com/ks/dist/child/sles12-sp4-updates-x86_64/sles12-sp4-x86_64/`). Activating *Ask On Error* ensures that you will be informed if a repository is not available during node deployment, otherwise errors will be silently ignored.

The *Add Repository* dialog allows adding additional repositories. See [Q:](#) for instructions.



Tip: Default Locations

If you have made the repositories available in the default locations on the Administration Server (see [Table 5.5, “Default Repository Locations on the Administration Server”](#) for a list), choose *Custom* and leave the *Repository URL* empty (default). The repositories will automatically be detected.

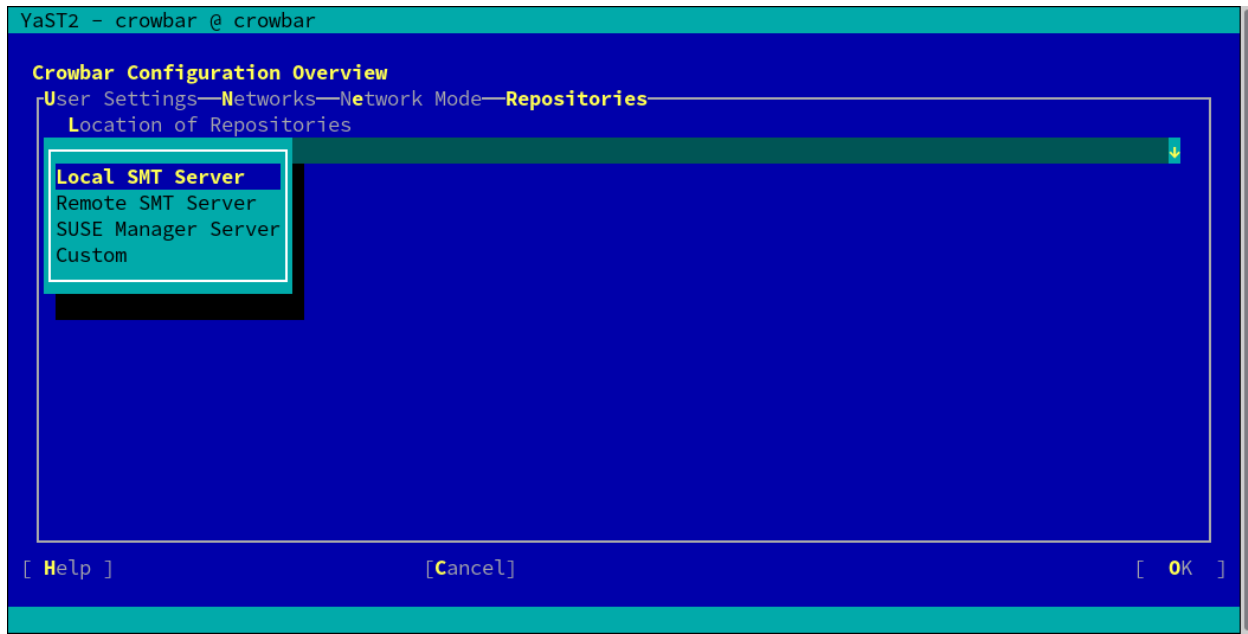


FIGURE 7.5: YAST CROWBAR SETUP: REPOSITORY SETTINGS

7.5 Custom Network Configuration

To adjust the pre-defined network setup of SUSE OpenStack Cloud beyond the scope of changing IP address assignments (as described in [Chapter 7, Crowbar Setup](#)), modify the network barclamp template.

The Crowbar network barclamp provides two functions for the system. The first is a common role to instantiate network interfaces on the Crowbar managed systems. The other function is address pool management. While the addresses can be managed with the YaST Crowbar module, complex network setups require to manually edit the network barclamp template file `/etc/crowbar/network.json`. This section explains the file in detail. Settings in this file are applied to all nodes in SUSE OpenStack Cloud. (See [Section 7.5.11, “Matching Logical and Physical Interface Names with network-json-resolve”](#) to learn how to verify your correct network interface names.)



Warning: No Network Changes After Completing the SUSE OpenStack Cloud Crowbar installation

After you have completed the SUSE OpenStack Cloud Crowbar installation, you cannot change the network setup. If you do need to change it, you must completely set up the Administration Server again.

The only exception to this rule is the interface map, which can be changed after setup. See [Section 7.5.3, “Interface Map”](#) for details.

7.5.1 Editing `network.json`

The `network.json` file is located in `/etc/crowbar/`. The template has the following general structure:

```
{
  "attributes" : {
    "network" : {
      "mode" : "VALUE",
      "start_up_delay" : VALUE,
      "teaming" : { "mode": VALUE }, ❶
      "enable_tx_offloading" : VALUE,
      "enable_rx_offloading" : VALUE,
      "interface_map" ❷ : [
        ...
      ],
      "conduit_map" ❸ : [
        ...
      ],
      "networks" ❹ : {
        ...
      },
    }
  }
}
```

- ❶ General attributes. Refer to [Section 7.5.2, “Global Attributes”](#) for details.
- ❷ Interface map section. Defines the order in which the physical network interfaces are to be used. Refer to [Section 7.5.3, “Interface Map”](#) for details.
- ❸ Network conduit section defining the network modes and the network interface usage. Refer to [Section 7.5.5, “Network Conduits”](#) for details.

- 4 Network definition section. Refer to [Section 7.5.7, “Network Definitions”](#) for details.



Note: Order of Elements

The order in which the entries in the `network.json` file appear may differ from the one listed above. Use your editor's search function to find certain entries.

7.5.2 Global Attributes

The most important options to define in the global attributes section are the default values for the network and bonding modes. The following global attributes exist:

```
{
  "attributes" : {
    "network" : {
      "mode" : "single", ❶
      "start_up_delay" : 30, ❷
      "teaming" : { "mode": 5 }, ❸
      "enable_tx_offloading" : true, ❹
      "enable_rx_offloading" : true, ❹
      "interface_map" : [
        ...
      ],
      "conduit_map" : [
        ...
      ],
      "networks" : {
        ...
      },
    }
  }
}
```

- ❶ Network mode. Defines the configuration name (or name space) to be used from the `conduit_map` (see [Section 7.5.5, “Network Conduits”](#)). Your choices are `single`, `dual`, or `team`.
- ❷ Time (in seconds) the Chef-client waits for the network interfaces to come online before timing out.
- ❸ Default bonding mode. For a list of available modes, see [Section 7.3, “Network Mode”](#).

- 4 Turn on/off TX and RX checksum offloading. If set to `false`, disable offloading by running `ethtool -K` and adding the setting to the respective ifcfg configuration file. If set to `true`, use the defaults of the network driver. If the network driver supports TX and/or RX checksum offloading and enables it by default, it will be used.

Checksum offloading is set to `true` in `network.json` by default. It is recommended to keep this setting. If you experience problems, such as package losses, try disabling this feature by setting the value to `false`.

Important: Change of the Default Value

Starting with SUSE OpenStack Cloud Crowbar, the default value for TX and RX checksum offloading changed from `false` to `true`.

To check which defaults a network driver uses, run `ethtool -k`, for example:

```
tux > sudo ethtool -k eth0 | grep checksumming
rx-checksumming: on
tx-checksumming: on
```

Note that if the output shows a value marked as `[fixed]`, this value cannot be changed. For more information on TX and RX checksum offloading refer to your hardware vendor's documentation. Detailed technical information can also be obtained from <https://www.kernel.org/doc/Documentation/networking/checksum-offloads.txt>.

7.5.3 Interface Map

By default, physical network interfaces are used in the order they appear under `/sys/class/net/`. If you want to apply a different order, you need to create an interface map where you can specify a custom order of the bus IDs. Interface maps are created for specific hardware configurations and are applied to all machines matching this configuration.

```
{
  "attributes" : {
    "network" : {
      "mode" : "single",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true ,
      "enable_rx_offloading" : true ,
    }
  }
}
```

```

    "interface_map" : [
      {
        "pattern" : "PowerEdge R610" ❶,
        "serial_number" : "0x02159F8E" ❷,
        "bus_order" : [ ❸
          "0000:00/0000:00:01",
          "0000:00/0000:00:03"
        ]
      }
      ...
    ],
    "conduit_map" : [
      ...
    ],
    "networks" : {
      ...
    }
  }
}

```

- ❶ Hardware specific identifier. This identifier can be obtained by running the command `dmidecode -s system-product-name` on the machine you want to identify. You can log in to a node during the hardware discovery phase (when booting the SLEShammer image) via the Administration Server.
- ❷ Additional hardware specific identifier. This identifier can be used in case two machines have the same value for *pattern*, but different interface maps are needed. Specifying this parameter is optional (it is not included in the default `network.json` file). The serial number of a machine can be obtained by running the command `dmidecode -s system-serial-number` on the machine you want to identify.
- ❸ Bus IDs of the interfaces. The order in which they are listed here defines the order in which Chef addresses the interfaces. The IDs can be obtained by listing the contents of `/sys/class/net/`.

❗ Important: PXE Boot Interface Must be Listed First

The physical interface used to boot the node via PXE must always be listed first.



Note: Interface Map Changes Allowed After Having Completed the SUSE OpenStack Cloud Crowbar Installation

Contrary to all other sections in `network.json`, you can change interface maps after completing the SUSE OpenStack Cloud Crowbar installation. However, nodes that are already deployed and affected by these changes must be deployed again. Therefore, we do not recommend making changes to the interface map that affect active nodes.

If you change the interface mappings after completing the SUSE OpenStack Cloud Crowbar installation you *must not* make your changes by editing `network.json`. You must rather use the Crowbar Web interface and open *Barclamps > Crowbar > Network > Edit*. Activate your changes by clicking *Apply*.

7.5.4 Interface Map Example

EXAMPLE 7.1: CHANGING THE NETWORK INTERFACE ORDER ON A MACHINE WITH FOUR NICs

1. Get the machine identifier by running the following command on the machine to which the map should be applied:

```
~ # dmidecode -s system-product-name
AS 2003R
```

The resulting string needs to be entered on the *pattern* line of the map. It is interpreted as a Ruby regular expression (see <http://www.ruby-doc.org/core-2.0/Regexp.html> for a reference). Unless the pattern starts with `^` and ends with `$`, a substring match is performed against the name returned from the above commands.

2. List the interface devices in `/sys/class/net` to get the current order and the bus ID of each interface:

```
~ # ls -lgG /sys/class/net/ | grep eth
lrwxrwxrwx 1 0 Jun 19 08:43 eth0 -> ../../devices/pci0000:00/0000:00:1c.0/0000:09:00.0/net/eth0
lrwxrwxrwx 1 0 Jun 19 08:43 eth1 -> ../../devices/pci0000:00/0000:00:1c.0/0000:09:00.1/net/eth1
lrwxrwxrwx 1 0 Jun 19 08:43 eth2 -> ../../devices/pci0000:00/0000:00:1c.0/0000:09:00.2/net/eth2
lrwxrwxrwx 1 0 Jun 19 08:43 eth3 -> ../../devices/pci0000:00/0000:00:1c.0/0000:09:00.3/net/eth3
```

The bus ID is included in the path of the link target—it is the following string: `../../../../devices/pciBUS ID/net/eth0`

3. Create an interface map with the bus ID listed in the order the interfaces should be used. Keep in mind that the interface from which the node is booted using PXE must be listed first. In the following example the default interface order has been changed to `eth0`, `eth2`, `eth1` and `eth3`.

```
{
  "attributes" : {
    "network" : {
      "mode" : "single",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true,
      "enable_rx_offloading" : true,
      "interface_map" : [
        {
          "pattern" : "AS 2003R",
          "bus_order" : [
            "0000:00/0000:00:1c.0/0000:09:00.0",
            "0000:00/0000:00:1c.0/0000:09:00.2",
            "0000:00/0000:00:1c.0/0000:09:00.1",
            "0000:00/0000:00:1c.0/0000:09:00.3"
          ]
        }
        ...
      ],
      "conduit_map" : [
        ...
      ],
      "networks" : {
        ...
      }
    }
  }
}
```

7.5.5 Network Conduits

Network conduits define mappings for logical interfaces—one or more physical interfaces bonded together. Each conduit can be identified by a unique name, the *pattern*. This pattern is also called “Network Mode” in this document.

Three network modes are available:

single: Only use the first interface for all networks. VLANs will be added on top of this single interface.

dual: Use the first interface as the admin interface and the second one for all other networks. VLANs will be added on top of the second interface.

team: Bond the first two or more interfaces. VLANs will be added on top of the bond.

See [Section 2.1.2, “Network Modes”](#) for detailed descriptions. Apart from these modes a fallback mode `".*/.*/.*"` is also pre-defined—it is applied in case no other mode matches the one specified in the global attributes section. These modes can be adjusted according to your needs. It is also possible to customize modes, but mode names must be either `single`, `dual`, or `team`. The mode name that is specified with `mode` in the global attributes section is deployed on all nodes in SUSE OpenStack Cloud. It is not possible to use a different mode for a certain node. However, you can define “sub” modes with the same name that only match the following machines:

- Machines with a certain number of physical network interfaces.
- Machines with certain roles (all Compute Nodes for example).

```
{
  "attributes" : {
    "network" : {
      "mode" : "single",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true,
      "enable_rx_offloading" : true,
      "interface_map" : [
        ...
      ],
      "conduit_map" : [
        {
          "pattern" : "single/.*/.*" ❶,
          "conduit_list" : {
            "intf2" ❷ : {
              "if_list" : ["lg1", "lg2"] ❸,
              "team_mode" : 5 ❹
            },
            "intf1" : {
              "if_list" : ["lg1", "lg2"],
              "team_mode" : 5
            }
          }
        }
      ]
    }
  }
}
```

```

        },
        "intf0" : {
            "if_list" : ["lg1","lg2"],
            "team_mode" : 5
        }
    }
},
...
],
"networks" : {
    ...
},
}
}
}
}

```

- ① This line contains the pattern definition for the `conduit_map`. The value for pattern must have the following form:

```
MODE_NAME/NUMBER_OF_NICS/NODE_ROLE
```

Each field in the pattern is interpreted as a Ruby regular expression (see <http://www.ruby-doc.org/core-2.0/Regexp.html> for a reference).

mode_name

Name of the network mode. This string is used to reference the mode from the general attributes section.

number_of_nics

Normally it is not possible to apply different network modes to different roles—you can only specify one mode in the global attributes section. However, it does not make sense to apply a network mode that bonds three interfaces on a machine with only two physical network interfaces. This option enables you to create modes for nodes with a given number of interfaces.

node_role

This part of the pattern lets you create matches for a certain node role. This enables you to create network modes for certain roles, for example the Compute Nodes (role: *nova-compute*) or the swift nodes (role: *swift-storage*). See [Example 7.3, “Network Modes for Certain Roles”](#) for the full list of roles.

- ② The logical network interface definition. Each conduit list must contain at least one such definition. This line defines the name of the logical interface. This identifier must be unique and will also be referenced in the network definition section. We recommend sticking with the pre-defined naming scheme: `intf0` for “Interface 0”, `intf1` for “Interface 1”, etc. If you change the name (not recommended), you also need to change all references in the network definition section.
- ③ This line maps one or more *physical* interfaces to the logical interface. Each entry represents a physical interface. If more than one entry exists, the interfaces are bonded—either with the mode defined in the `team_mode` attribute of this conduit section. Or, if that is not present, by the globally defined `teaming` attribute.

The physical interfaces definition needs to fit the following pattern:

```
[Quantifier][Speed][Order]
```

Valid examples are `+1g2`, `10g1` or `?1g2`.

Quantifier

Specifying the quantifier is optional. The following values may be entered:

`+`: at least the speed specified afterwards (specified value or higher)

`-`: at most the speed specified afterwards (specified value or lower)

`?`: any speed (speed specified afterwards is ignored)

If no quantifier is specified, the exact speed specified is used.

Speed

Specifying the interface speed is mandatory (even if using the `?` quantifier). The following values may be entered:

`10m`: 10 Mbit

`100m`: 100 Mbit

`1g`: 1 Gbit

`10g`: 10 Gbit

`20g`: 20 Gbit

`40g`: 40 Gbit

`56g`: 56 Gbit

Order

Position in the interface order. Specifying this value is mandatory. The interface order is defined by the order in which the interfaces appear in `/sys/class/net` (default) or, if it exists, by an interface map. The order is also linked to the speed in this context:

1g1: the first 1Gbit interface

+1g1: the first 1Gbit or 10Gbit interface. Crowbar will take the first 1Gbit interface. Only if such an interface does not exist, it will take the first 10Gbit interface available.

?1g3: the third 1Gbit, 10Gbit, 100Mbit or 10Mbit interface. Crowbar will take the third 1Gbit interface. Only if such an interface does not exist, it will take the third 10Gbit interface, then the third 100Mbit or 10Mbit interface.



Note: Ordering Numbers

Ordering numbers start with 1 rather than with 0.

Each interfaces that supports multiple speeds is referenced by multiple names—one for each speed it supports. A 10Gbit interface is therefore represented by four names: 10gX, 1gX, 100mX, 10mX, where X is the ordering number.

Ordering numbers always start with 1 and are assigned ascending for each speed, for example 1g1, 1g2, and 1g3. Numbering starts with the first physical interface. On systems with network interfaces supporting different maximum speeds, ordering numbers for the individual speeds differ, as the following example shows:

100Mbit (first interface): 100m1, 10m1

1Gbit (second interface): 1g1, 100m2, 10m2

10Gbit (third interface): 10g1, 1g2, 100m3, 10m3

In this example the pattern ?1g3 would match 100m3, since no third 1Gbit or 10Gbit interface exist.

- 4 The bonding mode to be used for this logical interface. Overwrites the default set in the global attributes section *for this interface*. See <https://www.kernel.org/doc/Documentation/networking/bonding.txt> for a list of available modes. Specifying this option is optional—if not specified here, the global setting applies.

7.5.6 Network Conduit Examples

EXAMPLE 7.2: NETWORK MODES FOR DIFFERENT NIC NUMBERS

The following example defines a team network mode for nodes with 6, 3, and an arbitrary number of network interfaces. Since the first mode that matches is applied, it is important that the specific modes (for 6 and 3 NICs) are listed before the general mode:

```
{
  "attributes" : {
    "network" : {
      "mode" : "single",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true,
      "enable_rx_offloading" : true,
      "interface_map" : [
        ...
      ],
      "conduit_map" : [
        {
          "pattern" : "single/6/*.*",
          "conduit_list" : {
            ...
          }
        },
        {
          "pattern" : "single/3/*.*",
          "conduit_list" : {
            ...
          }
        },
        {
          "pattern" : "single/*.*/.*",
          "conduit_list" : {
            ...
          }
        },
        ...
      ],
      "networks" : {
        ...
      },
    }
  }
}
```

EXAMPLE 7.3: NETWORK MODES FOR CERTAIN ROLES

The following example defines network modes for Compute Nodes with four physical interfaces, the Administration Server (role `crowbar`), the Control Node, and a general mode applying to all other nodes.

```
{
  "attributes" : {
    "network" : {
      "mode" : "team",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true,
      "enable_rx_offloading" : true,
      "interface_map" : [
        ...
      ],
      "conduit_map" : [
        {
          "pattern" : "team/4/nova-compute",
          "conduit_list" : {
            ...
          }
        },
        {
          "pattern" : "team/.*/^crowbar$",
          "conduit_list" : {
            ...
          }
        },
        {
          "pattern" : "team/.*/nova-controller",
          "conduit_list" : {
            ...
          }
        },
        {
          "pattern" : "team/.*/.*",
          "conduit_list" : {
            ...
          }
        },
        ...
      ],
      "networks" : {
        ...
      },
    },
  },
}
```

```
}  
}  
}
```

The following values for `node_role` can be used:

ceilometer-central
ceilometer-server
cinder-controller
cinder-volume
crowbar
database-server
glance-server
heat-server
horizon-server
keystone-server
manila-server
manila-share
monasca-agent
monasca-log-agent
monasca-master
monasca-server
neutron-network
neutron-server
nova-controller
nova-compute-*
rabbitmq-server
swift-dispersion
swift-proxy
swift-ring-compute
swift-storage

The role `crowbar` refers to the Administration Server.



Warning: The `crowbar` and Pattern Matching

As explained in [Example 7.4, "Network Modes for Certain Machines"](#), each node has an additional, unique role named `crowbar-FULLY_QUALIFIED_HOSTNAME`.

All three elements of the value of the `pattern` line are read as regular expressions. Therefore using the pattern `mode-name/.*crowbar` will match all nodes in your installation. `crowbar` is considered a substring and therefore will also match all strings `crowbar-FULLY QUALIFIED HOSTNAME`. As a consequence, all subsequent map definitions will be ignored. To make sure this does not happen, you must use the proper regular expression `^crowbar$: mode-name/.*^crowbar$`.

EXAMPLE 7.4: NETWORK MODES FOR CERTAIN MACHINES

Apart from the roles listed under [Example 7.3, “Network Modes for Certain Roles”](#), each node in SUSE OpenStack Cloud has a unique role, which lets you create modes matching exactly one node. Each node can be addressed by its unique role name in the `pattern` entry of the `conduit_map`.

The role name depends on the fully qualified host name (FQHN) of the respective machine. The role is named after the scheme `crowbar-FULLY QUALIFIED HOSTNAME` where colons are replaced with dashes, and periods are replaced with underscores. The FQHN depends on whether the respective node was booted via PXE or not.

To determine the host name of a node, log in to the Crowbar Web interface and go to `Nodes > Dashboard`. Click the respective node name to get detailed data for the node. The FQHN is listed first under *Full Name*.

Role Names for Nodes Booted via PXE

The `FULLY QUALIFIED HOSTNAME` for nodes booted via PXE is composed of the following: a prefix 'd', the MAC address of the network interface used to boot the node via PXE, and the domain name as configured on the Administration Server. A machine with the fully qualified host name `d1a-12-05-1e-35-49.cloud.example.com` would get the following role name:

```
crowbar-d1a-12-05-1e-35-49_cloud_example_com
```

Role Names for the Administration Server and Nodes Added Manually

The fully qualified hostnames of the Administration Server and all nodes added manually (as described in [Section 11.3, “Converting Existing SUSE Linux Enterprise Server 12 SP4 Machines Into SUSE OpenStack Cloud Nodes”](#)) are defined by the system administrator. They typically have the form `hostname + domain`, for example `admin.cloud.example.com`, which would result in the following role name:

```
crowbar-admin_cloud_example_com
```


Network mode definitions for certain machines must be listed first in the conduit map. This prevents other, general rules which would also map from being applied.

```
{
  "attributes" : {
    "network" : {
      "mode" : "dual",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true,
      "enable_rx_offloading" : true,
      "interface_map" : [
        ...
      ],
      "conduit_map" : [
        {
          "pattern" : "dual/.*/crowbar-d1a-12-05-1e-35-49_cloud_example_com",
          "conduit_list" : {
            ...
          }
        },
        ...
      ],
      "networks" : {
        ...
      },
    }
  }
}
```

7.5.7 Network Definitions

The network definitions contain IP address assignments, the bridge and VLAN setup, and settings for the router preference. Each network is also assigned to a logical interface defined in the network conduit section. In the following the network definition is explained using the example of the admin network definition:

```
{
  "attributes" : {
    "network" : {
      "mode" : "single",
      "start_up_delay" : 30,
      "teaming" : { "mode": 5 },
      "enable_tx_offloading" : true,
```


- ① Logical interface assignment. The interface must be defined in the network conduit section and must be part of the active network mode.
- ② Bridge setup. Do not touch. Should be `false` for all networks.
- ③ Create a VLAN for this network. Changing this setting is not recommended.
- ④ ID of the VLAN. Change this to the VLAN ID you intend to use for the specific network, if required. This setting can also be changed using the YaST Crowbar interface. The VLAN ID for the `nova-floating` network must always match the ID for the `public network`.
- ⑤ Router preference, used to set the default route. On nodes hosting multiple networks the router with the lowest `router_pref` becomes the default gateway. Changing this setting is not recommended.
- ⑥ Network address assignments. These values can also be changed by using the YaST Crowbar interface.
- ⑦ Openvswitch virtual switch setup. This attribute is maintained by Crowbar on a per-node level and should not be changed manually.
- ⑧ Name of the openvswitch virtual switch. This attribute is maintained by Crowbar on a per-node level and should not be changed manually.

! Important: VLAN Settings

As of SUSE OpenStack Cloud Crowbar 8, using a VLAN for the admin network is only supported on a native/untagged VLAN. If you need VLAN support for the admin network, it must be handled at switch level.

When changing the network configuration with YaST or by editing `/etc/crowbar/network.json`, you can define VLAN settings for each network. For the networks `nova-fixed` and `nova-floating`, however, special rules apply:

nova-fixed: The `USE VLAN` setting will be ignored. However, VLANs will automatically be used if deploying neutron with VLAN support (using the plugins `linuxbridge`, `openvswitch` plus VLAN, or `cisco` plus VLAN). In this case, you need to specify a correct `VLAN ID` for this network.

nova-floating: When using a VLAN for `nova-floating` (which is the default), the `USE VLAN` and `VLAN ID` settings for `nova-floating` and `public` default to the same.

You have the option of separating public and floating networks with a custom configuration. Configure your own separate floating network (not as a subnet of the public network), and give the floating network its own router. For example, define `nova-floating` as part of an external network with a custom `bridge-name`. When you are using different networks and OpenVSwitch is configured, the pre-defined `bridge-name` won't work.

7.5.8 Providing Access to External Networks

By default, external networks cannot be reached from nodes in the SUSE OpenStack Cloud. To access external services such as a SUSE Manager server, an SMT server, or a SAN, you need to make the external network(s) known to SUSE OpenStack Cloud. Do so by adding a network definition for each external network to `/etc/crowbar/network.json`. Refer to [Section 7.5, "Custom Network Configuration"](#) for setup instructions.

EXAMPLE 7.5: EXAMPLE NETWORK DEFINITION FOR THE EXTERNAL NETWORK 192.168.150.0/16

```
"external" : {
  "add_bridge" : false,
  "vlan" : XXX,
  "ranges" : {
    "host" : {
      "start" : "192.168.150.1",
      "end" : "192.168.150.254"
    }
  },
  "broadcast" : "192.168.150.255",
  "netmask" : "255.255.255.0",
  "conduit" : "intf1",
  "subnet" : "192.168.150.0",
  "use_vlan" : true
}
```

Replace the value `XXX` for the VLAN by a value not used within the SUSE OpenStack Cloud network and not used by neutron. By default, the following VLANs are already used:

TABLE 7.3: VLANS USED BY THE SUSE OPENSTACK CLOUD DEFAULT NETWORK SETUP

VLAN ID	Used by
100	BMC VLAN (bmc_vlan)

VLAN ID	Used by
200	Storage Network
300	Public Network (nova-floating, public)
400	Software-defined network (os_sdn)
500	Private Network (nova-fixed)
501 - 2500	neutron (value of nova-fixed plus 2000)

7.5.9 Split Public and Floating Networks on Different VLANs

For custom setups, the public and floating networks can be separated. Configure your own separate floating network (not as a subnet of the public network), and give the floating network its own router. For example, define `nova-floating` as part of an external network with a custom `bridge-name`. When you are using different networks and OpenVSwitch is configured, the pre-defined `bridge-name` won't work.

7.5.10 Adjusting the Maximum Transmission Unit for the Admin and Storage Network

If you need to adjust the Maximum Transmission Unit (MTU) for the Admin and/or Storage Network, adjust `/etc/crowbar/network.json` as shown below. You can also enable jumbo frames this way by setting the MTU to 9000. The following example enables jumbo frames for both, the storage and the admin network by setting `"mtu": 9000`.

```

"admin": {
  "add_bridge": false,
  "broadcast": "192.168.124.255",
  "conduit": "intf0",
  "mtu": 9000,
  "netmask": "255.255.255.0",
  "ranges": {
    "admin": {
      "end": "192.168.124.11",
      "start": "192.168.124.10"
    }
  },

```

```

    "dhcp": {
      "end": "192.168.124.80",
      "start": "192.168.124.21"
    },
    "host": {
      "end": "192.168.124.160",
      "start": "192.168.124.81"
    },
    "switch": {
      "end": "192.168.124.250",
      "start": "192.168.124.241"
    }
  },
  "router": "192.168.124.1",
  "router_pref": 10,
  "subnet": "192.168.124.0",
  "use_vlan": false,
  "vlan": 100
},
"storage": {
  "add_bridge": false,
  "broadcast": "192.168.125.255",
  "conduit": "intf1",
  "mtu": 9000,
  "netmask": "255.255.255.0",
  "ranges": {
    "host": {
      "end": "192.168.125.239",
      "start": "192.168.125.10"
    }
  },
  "subnet": "192.168.125.0",
  "use_vlan": true,
  "vlan": 200
},

```



Warning: No Network Changes After Completing the SUSE OpenStack Cloud Crowbar installation

After you have completed the SUSE OpenStack Cloud Crowbar installation, you cannot change the network setup, and you cannot change the MTU size.

7.5.11 Matching Logical and Physical Interface Names with `network-json-resolve`

SUSE OpenStack Cloud includes a new script, `network-json-resolve`, which matches the physical and logical names of network interfaces, and prints them to stdout. Use this to verify that you are using the correct interface names in `network.json`. Note that it will only work if OpenStack nodes have been deployed. The following command prints a help menu:

```
sudo /opt/dell/bin/network-json-resolve -h
```

`network-json-resolve` reads your deployed `network.json` file. To use a different `network.json` file, specify its full path with the `--network-json` option. The following example shows how to use a different `network.json` file, and prints the interface mappings of a single node:

```
sudo /opt/dell/bin/network-json-resolve --network-json /opt/configs/network.json aliases
computel
eth0: 0g1, 1g1
eth1: 0g1, 1g1
```

You may query the mappings of a specific network interface:

```
sudo /opt/dell/bin/network-json-resolve aliases computel eth0
eth0: 0g1, 1g1
```

Print the bus ID order on a node. This returns `no bus order defined for node` if you did not configure any bus ID mappings:

```
sudo /opt/dell/bin/network-json-resolve bus_order computel
```

Print the defined conduit map for the node:

```
sudo /opt/dell/bin/network-json-resolve conduit_map computel
bastion: ?1g1
intf0: ?1g1
intf1: ?1g1
intf2: ?1g1
```

Resolve conduits to the standard interface names:

```
sudo /opt/dell/bin/network-json-resolve conduits computel
bastion:
intf0: eth0
intf1: eth0
```

```
intf2: eth0
```

Resolve the configured networks on a node to the standard interface names:

```
sudo /opt/dell/bin/network-json-resolve networks compute1
bastion:
bmc_vlan: eth0
nova_fixed: eth0
nova_floating: eth0
os_sdn: eth0
public: eth0
storage: eth0
```

Resolve the specified network to the standard interface name(s):

```
sudo /opt/dell/bin/network-json-resolve networks compute1 public
public: eth0
```

Resolve a network.json-style interface to its standard interface name(s):

```
sudo /opt/dell/bin/network-json-resolve resolve compute1 lg1
eth0
```


8 Starting the SUSE OpenStack Cloud Crowbar installation

The last step in configuring the Administration Server is starting Crowbar.

Before starting the SUSE OpenStack Cloud Crowbar installation to finish the configuration of the Administration Server, make sure to double-check the following items.

FINAL CHECK POINTS

- Make sure the network configuration is correct. Run `YaST > Crowbar` to review/change the configuration. See [Chapter 7, Crowbar Setup](#) for further instructions.



Important: An HA Setup Requires Team Network Mode

If you are planning to make SUSE OpenStack Cloud highly available, whether upon the initial setup or later, set up the network in the team mode. Such a setup requires at least two network cards for each node.

- Make sure `hostname -f` returns a fully qualified host name. See [Chapter 6, Service Configuration: Administration Server Network Configuration](#) for further instructions.
- Make sure all update and product repositories are available. See [Chapter 5, Software Repository Setup](#) for further instructions.
- Make sure the operating system and SUSE OpenStack Cloud Crowbar are up-to-date and have the latest patches installed. Run `zypper patch` to install them.
- To use the Web interface for the SUSE OpenStack Cloud Crowbar installation you need network access to the Administration Server via a second network interface. As the network will be reconfigured during the SUSE OpenStack Cloud Crowbar installation, make sure to either have a bastion network or an external gateway configured. (For details on bastion networks, see [Section 7.3.1, “Setting Up a Bastion Network”](#).)

Now everything is in place to finally set up Crowbar and install the Administration Server. Crowbar requires a MariaDB database—you can either create one on the Administration Server or use an existing PostgreSQL database on a remote server.

PROCEDURE 8.1: SETTING UP CROWBAR WITH A LOCAL DATABASE

1. Start Crowbar:

```
sudo systemctl start crowbar-init
```

2. Create a new database on the Administration Server. By default the credentials `crowbar/crowbar` are used:

```
crowbarctl database create
```

To use a different user name and password, run the following command instead:

```
crowbarctl database create \  
--db_username=USERNAME --db_password=PASSWORD
```

Run `crowbarctl database help create` for help and more information.

PROCEDURE 8.2: SETTING UP CROWBAR WITH A REMOTE MARIADB DATABASE

1. Start Crowbar:

```
sudo systemctl start crowbar-init
```

2. Make sure a user account that can be used for the Crowbar database exists on the remote MariaDB database. If not, create such an account.
3. Test the database connection using the credentials from the previous step:

```
crowbarctl database test --db-username=USERNAME \  
--db-password=PASSWORD --database=DBNAME \  
--host=IP_or_FQDN --port=PORT
```

You need to be able to successfully connect to the database before you can proceed. Run `crowbarctl database help test` for help and more information.

4. To connect to the database, use the following command:

```
crowbarctl database connect --db-username=USERNAME \  
--db-password=PASSWORD --database=DBNAME \  
--host=IP_or_FQDN --port=PORT
```

Run `crowbarctl database help connect` for help and more information.

After the database is successfully created and you can connect to it, access the Web interface from a Web browser, using the following address:

```
http://ADDRESS
```

Replace ADDRESS either with the IP address of the second network interface or its associated host name. Logging in to the Web interface requires the credentials you configured with YaST Crowbar (see [Section 7.1, "User Settings"](#)). If you have not changed the defaults, user name and password are both crowbar. Refer to [Chapter 10, The Crowbar Web Interface](#) for details.

The Web interface shows the SUSE OpenStack Cloud installation wizard. Click *Start Installation* to begin. The installation progress is shown in the Web interface:

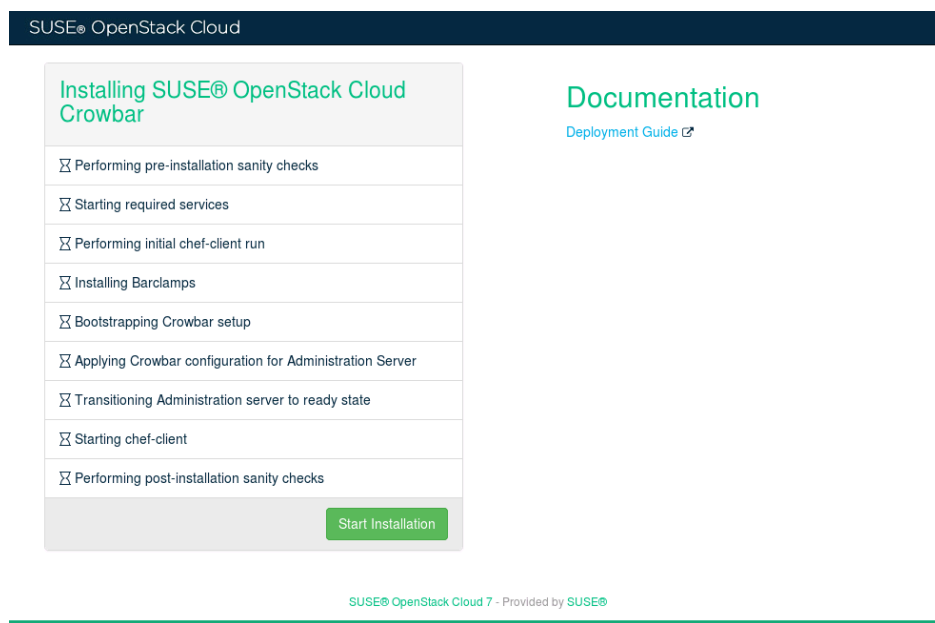


FIGURE 8.1: THE SUSE OPENSTACK CLOUD CROWBAR INSTALLATION WEB INTERFACE

If the installation has successfully finished, you will be redirected to the Crowbar Dashboard:

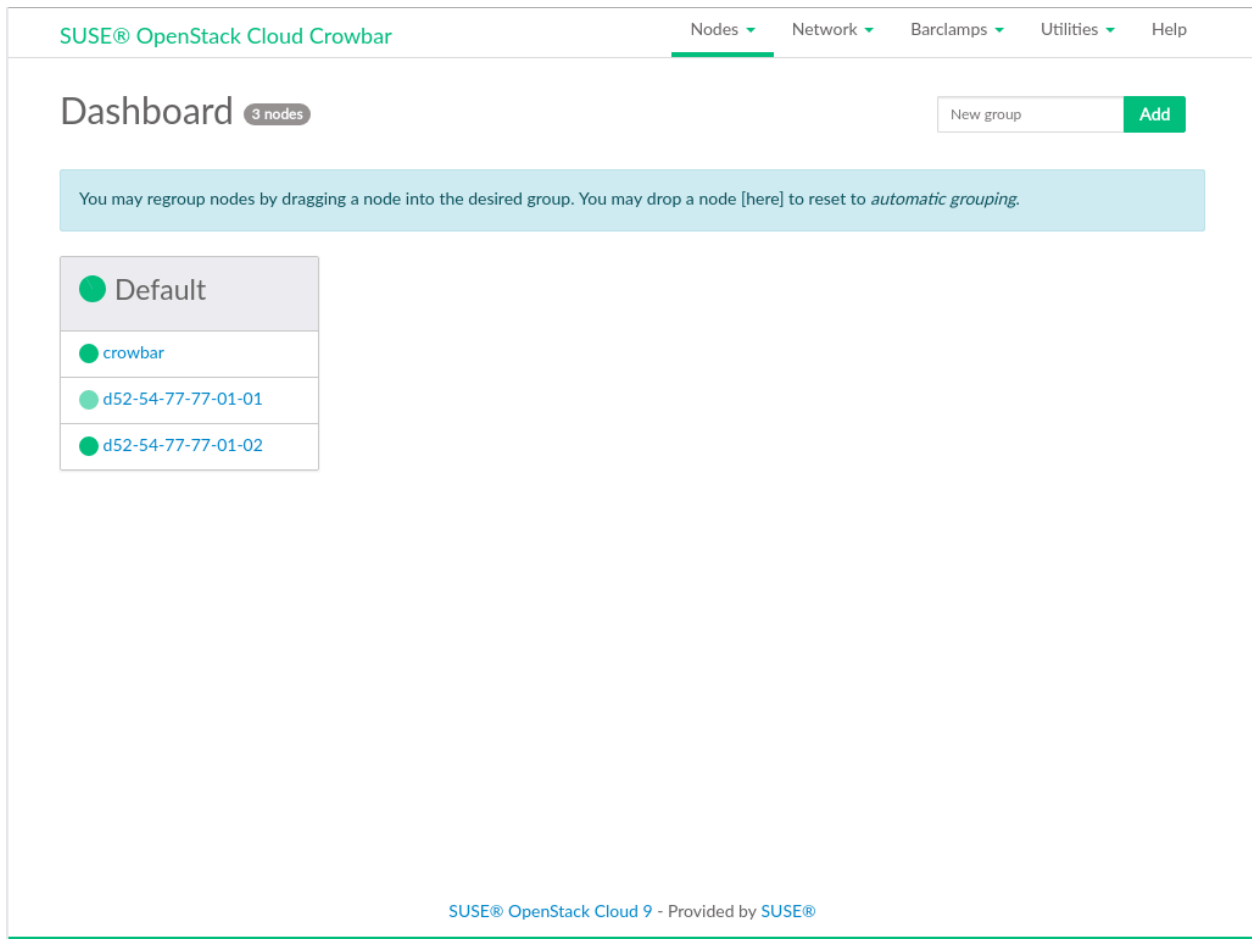


FIGURE 8.2: CROWBAR WEB INTERFACE: THE DASHBOARD

From here you can start allocating nodes and then deploy the OpenStack services. Refer to [Part III, "Setting Up OpenStack Nodes and Services"](#) for more information.

9 Customizing Crowbar

9.1 Skip Unready Nodes

In large deployments with many nodes, there are always some nodes that are in a fail or unknown state. New barclamps cannot be applied to them and values cannot be updated in some barclamps that are already deployed. This happens because Crowbar will refuse to apply a barclamp to a list of nodes if they are not all in `ready` state.

To avoid having to manually take out nodes that are not `ready`, there is a feature called `skip_unready_nodes`. Instead of refusing to apply the barclamp, it will skip the nodes that it finds in any other state than `ready`.

Enabling the Feature

In `/opt/dell/crowbar_framework/config/crowbar.yml`, set the option `skip_unready_nodes` to `true`.

```
default: &default
skip_unready_nodes:
  enabled: false <<< change to true
```

Roles Affected

All Barclamp roles are affected. The default config file includes all the roles that have been tested and found to be working. Adding roles to the default list is not supported for the `skip_unready_nodes` feature. Removing default roles is supported.

The list of currently supported roles to skip:

```
- bmc-nat-client
- ceilometer-agent
- deployer-client
- dns-client
- ipmi
- logging-client
- nova-compute-ironic
- nova-compute-kvm
- nova-compute-qemu
- nova-compute-vmware
- ntp-client
- provisioner-base
- suse-manager-client
- swift-storage
```

```
- updater
```

Determining Which Nodes Were Skipped

Skipped nodes are logged to the Crowbar log (`/var/log/crowbar/production.log`) where you can search for the text `skipped until next chef run`. This will print the log lines where nodes were skipped, the name of the node, and the barclamp which was being applied.

```
tux > grep "skipped until next chef run" /var/log/crowbar/production.log
```

! Important

After enabling/disabling the `skip_unready_nodes` feature or adding/removing roles, the Crowbar framework service must be restarted (`systemctl restart crowbar`) in order to use the updated settings.

9.2 Skip Unchanged Nodes

When a barclamp is applied, all nodes will run `chef-client`. Sometimes it is not necessary to run chef for each node as attributes or roles may have not changed for some of the nodes.

The `skip_unchanged_nodes` feature allows nodes to be skipped if their attributes or roles have not changed. This can help speed up applying a barclamp, especially in large deployments and with roles that are usually applied to a lot of nodes, such as nova.

Enabling the Feature

In `/opt/dell/crowbar_framework/config/crowbar.yml`, set the option `skip_unchanged_nodes` to `true`.

```
default: &default
skip_unchanged_nodes:
  enabled: false <<< change to true
```

9.3 Controlling Chef Restarts Manually

When a service configuration has changed, Chef forces this service to restart. Sometimes it is useful to have manual control of these restarts. This feature supports avoiding automatic restart of services and including them in a pending restart list. Disabling restarts is enabled/disabled by barclamp and cannot be done on a service level. In other words, enabling this feature on the cinder barclamp will disable automatic restarts for all cinder-* services.

Two steps are necessary to activate this feature:

1. Enable `disallow_restart` in the Crowbar configuration file
2. Set the `disable_restart` flag for a specific barclamp using `crowbar-client` or API

Enabling the Feature

1. In `/opt/dell/crowbar_framework/config/crowbar.yml`, set the option `disallow_restart` to `true`.

```
default: &default
disallow_restart:
  enabled: false <<< change to true
```

2. The `disable_restart` flag can be set with the Crowbar client or with the API.

- Set Flag with Crowbar Client

The crowbar client options for this feature are accessed with the `crowbarctl services` command, and only work for OpenStack services. The options are:

`disable_restart`

The parameters are the barclamp name and the flag value (`true` to disable automatic restart and `false` to enable automatic restart). The command is `crowbarctl services disable_restart BARCLAMP <true|false>`. For example, to disable restart of the keystone barclamp, enter the command `crowbarctl services disable_restart keystone true`.

`restart_flags`

Used to check the `disable_restart` flag for each barclamp

```
crowbarctl services restart_flags
{
  "nova": true,
  "keystone": true,
```

```
"database": true
}
```

list_restarts

Displays the list of pending restarts. The `pending_restart` flag indicates that a service tried to restart due to the Chef run but it did not due to the automatic restart being disabled. It also indicates that the service might have a new configuration and it will not be applied until it is manually restarted.

In the following example, the `pacemaker_service` attribute indicates whether this service is managed by Pacemaker (usually in an HA environment) or it is a standalone service managed by `systemd` (usually in non-HA environments). More on Pacemaker at [Section 12.2, "Deploying Pacemaker \(Optional, HA Setup Only\)"](#). The service to restart will be `apache2` not managed by Pacemaker.

```
crowbarctl services list_restarts
{
  "NODE_IP": {
    "alias": "controller1",
    "keystone": {
      "apache2": {
        "pacemaker_service": false,
        "timestamp": "2017-11-22 11:17:49 UTC"
      }
    }
  }
}
```

clear_restart

Removes the flag on a specific node for a specific service. It can be executed when the service has restarted manually. The command is `crowbarctl services clear_restart NODE SERVICE`. For example, `crowbarctl services clear_restart NODE_IP apache2`

- Using the API to List, Get Status, Set and Clear Flags
 - `/restart_management/configuration`

GET

Lists the barclamps and the status of service reboots disallowed

POST (parameters: `disallow_restarts`, `barclamp`)
Sets the `disallow_restart` flag for a barclamp

- `/restart_management/restarts`

GET
Lists all of the services needing restarts

POST (parameters: `node`, `service`)
Clears the restart flag for the given service in the given node

9.4 Prevent Automatic Restart

Sometimes a change in a proposal requires services to be restarted on all implicated nodes. This could be a problem in a production environment where interrupting a service completely is not an option, and where manual restart control is needed. With the service `disable_restart` feature in `crowbarctl`, you can set a flag for certain proposals to prevent the automatic restart.

Enabling the Feature

1. In `/opt/dell/crowbar_framework/config/crowbar.yml`, set the option `disallow_restart` to `true`.

```
default: &default
disable_restart:
  enabled: false <<< change to true
```

2. Restart Crowbar
3. Enable the `disable_restart` flag in the affected cookbook.

```
crowbarctl services disable_restart COOKBOOK true
```

For example, to disable keystone and RabbitMQ restarts:

```
crowbarctl services disable_restart keystone true
crowbarctl services disable_restart rabbitmq true
```

4. To check the proposals that have `disable_restart` enabled:

```
crowbarctl services restart_flags
{
```

```
"rabbitmq": true,  
"keystone": true,  
}
```

Check Pending Restarts

When a proposal is applied with the `disable_restart` flag enabled, the implicated nodes will not restart. They will be listed as pending restarts. To check this list, run the command `crowbarctl services list_restarts`.

In the following example, `disable_restart` is enabled for RabbitMQ. After applying `rabbitmq`, `list_restarts` will show the affected nodes, the proposal, and the services to restart.

```
crowbarctl services list_restarts  
{  
  "d52-54-77-77-01-01.vo5.cloud.suse.de": {  
    "alias": "controller1",  
    "rabbitmq": {  
      "rabbitmq-server": {  
        "pacemaker_service": false,  
        "timestamp": "2018-03-07 15:30:30 UTC"  
      }  
    }  
  }  
}
```

After a manual restart of the service, the flags should be cleaned. The following command will clean the flag for a specific node, for a specific proposal or all proposals, and for a specific service.

```
crowbarctl services clear_restart NODE [COOKBOOK [SERVICE]]
```

For example, to clean the RabbitMQ flag from the `controller1` node, run the command:

```
crowbarctl service clear_restart controller1 rabbitmq
```

To clean the `controller1` node, run the command:

```
crowbarctl service clear_restart controller1
```

III Setting Up OpenStack Nodes and Services

- 10 The Crowbar Web Interface **108**
- 11 Installing the OpenStack Nodes **119**
- 12 Deploying the OpenStack Services **143**
- 13 Limiting Users' Access Rights **271**
- 14 Configuration Files for OpenStack Services **277**
- 15 Installing SUSE CaaS Platform heat Templates **280**
- 16 Installing SUSE CaaS Platform v4 using terraform **291**

10 The Crowbar Web Interface

The Crowbar Web interface runs on the Administration Server. It provides an overview of the most important deployment details in your cloud. This includes a view of the nodes and which roles are deployed on which nodes, and the barclamp proposals that can be edited and deployed. In addition, the Crowbar Web interface shows details about the networks and switches in your cloud. It also provides graphical access to tools for managing your repositories, backing up or restoring the Administration Server, exporting the Chef configuration, or generating a `supportconfig` TAR archive with the most important log files.



Tip: Crowbar API Documentation

You can access the Crowbar API documentation from the following static page: http://CROWBAR_SERVER/apidoc.

The documentation contains information about the crowbar API endpoints and its parameters, including response examples, possible errors (and their HTTP response codes), parameter validations, and required headers.

10.1 Logging In

The Crowbar Web interface uses the HTTP protocol and port `80`.

PROCEDURE 10.1: LOGGING IN TO THE CROWBAR WEB INTERFACE

1. On any machine, start a Web browser and make sure that JavaScript and cookies are enabled.
2. As URL, enter the IP address of the Administration Server, for example:

```
http://192.168.124.10/
```
3. Log in as user `crowbar`. If you have not changed the password, it is `crowbar` by default.

1. On the Administration Server, open the following file in a text editor: `/etc/crowbarrc`. It contains the following:

```
[default]
username=crowbar
password=crowbar
```

Change the `password` entry and save the file.

2. Alternatively, use the YaST Crowbar module to edit the password as described in [Section 7.1, "User Settings"](#).
3. Manually run `chef-client`. This step is not needed if the installation has not been completed yet.

10.2 Overview: Main Elements

After logging in to Crowbar, you will see a navigation bar at the top-level row. Its menus and the respective views are described in the following sections.

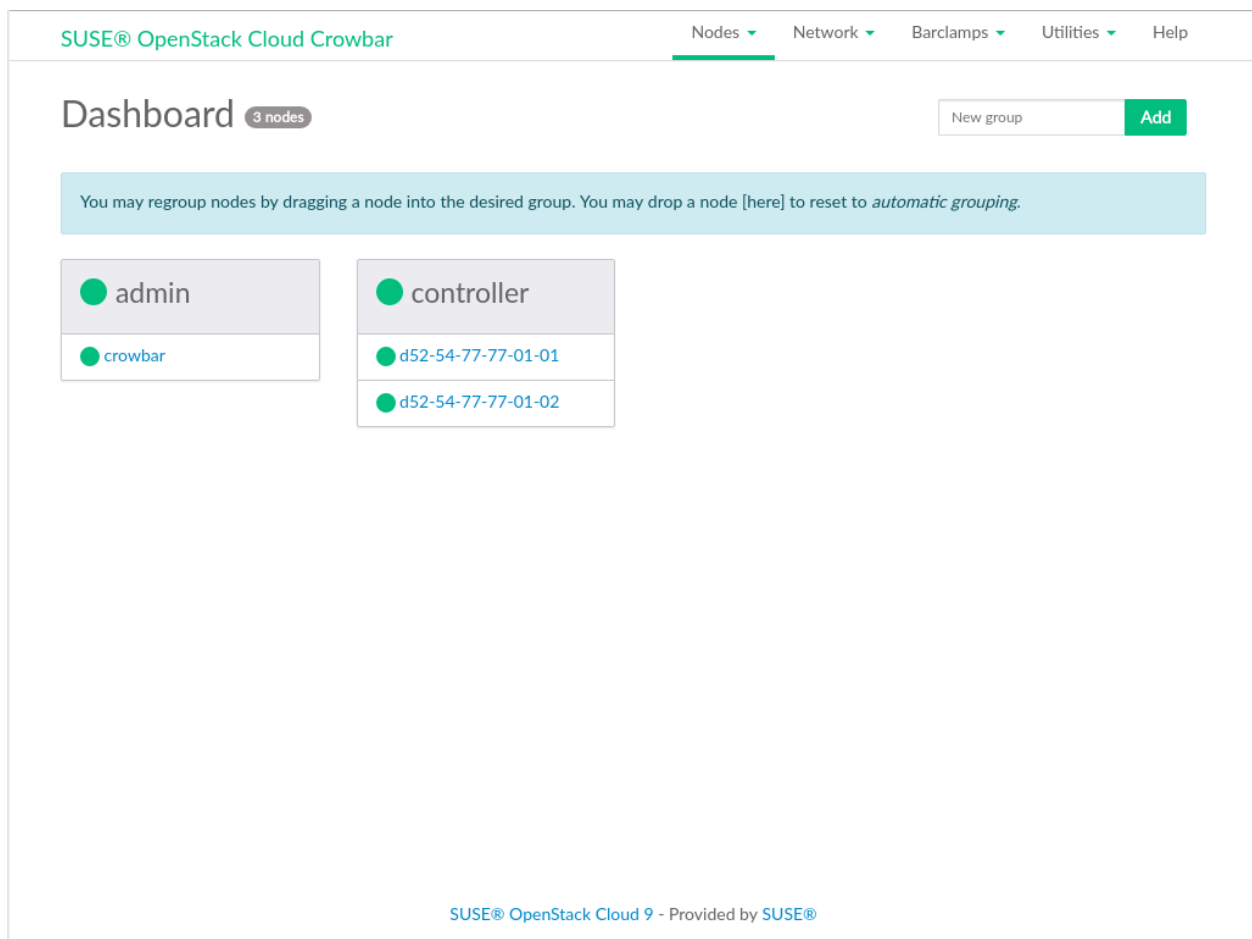


FIGURE 10.1: CROWBAR UI—DASHBOARD (MAIN SCREEN)

10.2.1 Nodes

Dashboard

This is the default view after logging in to the Crowbar Web interface. The Dashboard shows the groups (which you can create to arrange nodes according to their purpose), which nodes belong to each group, and which state the nodes and groups are in. In addition, the total number of nodes is displayed in the top-level row.

The color of the dot in front of each node or group indicates the status. If the dot for a group shows more than one color, hover the mouse pointer over the dot to view the total number of nodes and the statuses they are in.

- Gray means the node is being discovered by the Administration Server, or that there is no up-to-date information about a deployed node. If the status is shown for a node longer than expected, check if the chef-client is still running on the node.
- Yellow means the node has been successfully Discovered. As long as the node has not been allocated the dot will flash. A solid (non-flashing) yellow dot indicates that the node has been allocated, but installation has not yet started.
- Flashing from yellow to green means the node has been allocated and is currently being installed.
- Solid green means the node is in status Ready.
- Red means the node is in status Problem.

During the initial state of the setup, the Dashboard only shows one group called sw_unknown into which the Administration Server is automatically sorted. Initially, all nodes (except the Administration Server) are listed with their MAC address as a name. However, we recommend creating an alias for each node. This makes it easier to identify the node in the admin network and on the Dashboard. For details on how to create groups, how to assign nodes to a group, and how to create node aliases, see [Section 11.2, "Node Installation"](#).

Bulk Edit

This screen allows you to edit multiple nodes at once instead of editing them individually. It lists all nodes, including *Name* (in form of the MAC address), *Hardware* configuration, *Alias* (used within the admin network), *Public Name* (name used outside of the SUSE OpenStack Cloud network), *Group*, *Intended Role*, *Platform* (the operating system that is going to be installed on the node), *License* (if available), and allocation status. You can toggle the list view between *Show unallocated* or *Show all* nodes.

For details on how to fill in the data for all nodes and how to start the installation process, see [Section 11.2, "Node Installation"](#).

HA Clusters

This menu entry only appears if your cloud contains a High Availability setup. The overview shows all clusters in your setup, including the *Nodes* that are members of the respective cluster and the *Roles* assigned to the cluster. It also shows if a cluster contains *Remote Nodes* and which roles are assigned to the remote nodes.

Actives Roles

This overview shows which roles have been deployed on which node(s). The roles are grouped according to the service to which they belong. You cannot edit anything here. To change role deployment, you need to edit and redeploy the appropriate barclamps as described in [Chapter 12, Deploying the OpenStack Services](#).

10.2.2 Barclamps

All Barclamps

This screen shows a list of all available barclamp proposals, including their *Status*, *Name*, and a short *Description*. From here, you can *Edit* individual barclamp proposals as described in [Section 10.3, "Deploying Barclamp Proposals"](#).

Crowbar

This screen only shows the barclamps that are included with the core Crowbar framework. They contain general recipes for setting up and configuring all nodes. From here, you can *Edit* individual barclamp proposals.

OpenStack

This screen only shows the barclamps that are dedicated to OpenStack service deployment and configuration. From here, you can *Edit* individual barclamp proposals.

Deployment Queue

If barclamps are applied to one or more nodes that are not yet available for deployment (for example, because they are rebooting or have not been fully installed yet), the proposals will be put in a queue. This screen shows the proposals that are *Currently deploying* or *Waiting in queue*.

10.2.3 Utilities

Exported Items

The *Exported Files* screen allows you to export the Chef configuration and the `support-config` TAR archive. The `supportconfig` archive contains system information such as the current kernel version being used, the hardware, RPM database, partitions, and the most important log files for analysis of any problems. To access the export options, click *New Export*. After the export has been successfully finished, the *Exported Files* screen will show any files that are available for download.

Repositories

This screen shows an overview of the mandatory, recommended, and optional repositories for all architectures of SUSE OpenStack Cloud Crowbar. On each reload of the screen the Crowbar Web interface checks the availability and status of the repositories. If a mandatory repository is not present, it is marked red in the screen. Any repositories marked green are usable and available to each node in the cloud. Usually, the available repositories are also shown as *Active* in the rightmost column. This means that the managed nodes will automatically be configured to use this repository. If you disable the *Active* check box for a repository, managed nodes will not use that repository.

You cannot edit any repositories in this screen. If you need additional, third-party repositories, or want to modify the repository metadata, edit `/etc/crowbar/repos.yml`. Find an example of a repository definition below:

```
suse-12.2:
  x86_64:
    Custom-Repo-12.2:
      url: 'http://example.com/12-SP2:/x86_64/custom-repo/'
      ask_on_error: true # sets the ask_on_error flag in
                      # the autoyast profile for that repo
      priority: 99 # sets the repo priority for zypper
```

Alternatively, use the YaST Crowbar module to add or edit repositories as described in [Section 7.4, "Repositories"](#).

swift Dashboard

This screen allows you to run `swift-dispersion-report` on the node or nodes to which it has been deployed. Use this tool to measure the overall health of the swift cluster. For details, see <http://docs.openstack.org/liberty/config-reference/content/object-storage-dispersion.html>.

Backup & Restore

This screen is for creating and downloading a backup of the Administration Server. You can also restore from a backup or upload a backup image from your local file system.

10.2.4 Help

From this screen you can access HTML and PDF versions of the SUSE OpenStack Cloud Crowbar manuals that are installed on the Administration Server.

10.3 Deploying Barclamp Proposals

Barclamps are a set of recipes, templates, and installation instructions. They are used to automatically install OpenStack components on the nodes. Each barclamp is configured via a so-called proposal. A proposal contains the configuration of the service(s) associated with the barclamp and a list of machines onto which to deploy the barclamp.

Most barclamps consist of two sections:

Attributes

For changing the barclamp's configuration, either by editing the respective Web forms (*Custom* view) or by switching to the *Raw* view, which exposes all configuration options for the barclamp. In the *Raw* view, you directly edit the configuration file.



Important: Saving Your Changes

Before you switch to *Raw* view or back again to *Custom* view, *Save* your changes. Otherwise they will be lost.

Deployment

Lets you choose onto which nodes to deploy the barclamp. On the left-hand side, you see a list of *Available Nodes*. The right-hand side shows a list of roles that belong to the barclamp. Assign the nodes to the roles that should be deployed on that node. Some barclamps contain roles that can also be deployed to a cluster. If you have deployed the Pacemaker barclamp, the *Deployment* section additionally lists *Available Clusters* and *Available Clusters with Remote Nodes* in this case. The latter are clusters that contain both “normal” nodes and Pacemaker remote nodes. See [Section 2.6.3, “High Availability of the Compute Node\(s\)”](#) for the basic details.

Important: Clusters with Remote Nodes

- Clusters (or clusters with remote nodes) cannot be assigned to roles that need to be deployed on individual nodes. If you try to do so, the Crowbar Web interface shows an error message.
- If you assign a cluster with remote nodes to a role that can only be applied to “normal” (Corosync) nodes, the role will only be applied to the Corosync nodes of that cluster. The role will not be applied to the remote nodes of the same cluster.

10.3.1 Creating, Editing and Deploying Barclamp Proposals

The following procedure shows how to generally edit, create and deploy barclamp proposals. For the description and deployment of the individual barclamps, see [Chapter 12, Deploying the OpenStack Services](#).

1. Log in to the Crowbar Web interface.
2. Click *Barclamps* and select *All Barclamps*. Alternatively, filter for categories by selecting either *Crowbar* or *OpenStack*.
3. To create a new proposal or edit an existing one, click *Create* or *Edit* next to the appropriate barclamp.
4. Change the configuration in the *Attributes* section:
 - a. Change the available options via the Web form.
 - b. To edit the configuration file directly, first save changes made in the Web form. Click *Raw* to edit the configuration in the editor view.
 - c. After you have finished, *Save* your changes. (They are not applied yet).
5. Assign nodes to a role in the *Deployment* section of the barclamp. By default, one or more nodes are automatically pre-selected for available roles.
 - a. If this pre-selection does not meet your requirements, click the *Remove* icon next to the role to remove the assignment.

- b. To assign a node or cluster of your choice, select the item you want from the list of nodes or clusters on the left-hand side, then drag and drop the item onto the desired role name on the right.



Note

Do *not* drop a node or cluster onto the text box—this is used to filter the list of available nodes or clusters!

- c. To save your changes without deploying them yet, click *Save*.

6. Deploy the proposal by clicking *Apply*.



Warning: Wait Until a Proposal Has Been Deployed

If you deploy a proposal onto a node where a previous one is still active, the new proposal will overwrite the old one.

Deploying a proposal might take some time (up to several minutes). Always wait until you see the message “Successfully applied the proposal” before proceeding to the next proposal.

A proposal that has not been deployed yet can be deleted in the *Edit Proposal* view by clicking *Delete*. To delete a proposal that has already been deployed, see [Section 10.3.3, “Deleting a Proposal That Already Has Been Deployed”](#).

10.3.2 Barclamp Deployment Failure



Warning: Deployment Failure

A deployment failure of a barclamp may leave your node in an inconsistent state. If deployment of a barclamp fails:

1. Fix the reason that has caused the failure.
2. Re-deploy the barclamp.

For help, see the respective troubleshooting section at [Q & A 2, "OpenStack Node Deployment"](#).

10.3.3 Deleting a Proposal That Already Has Been Deployed

To delete a proposal that has already been deployed, you first need to *Deactivate* it.

PROCEDURE 10.3: DEACTIVATING AND DELETING A PROPOSAL

1. Log in to the Crowbar Web interface.
2. Click *Barclamps* > *All Barclamps*.
3. Click *Edit* to open the editing view.
4. Click *Deactivate* and confirm your choice in the following pop-up.
Deactivating a proposal removes the chef role from the nodes, so the routine that installed and set up the services is not executed anymore.
5. Click *Delete* to confirm your choice in the following pop-up.
This removes the barclamp configuration data from the server.

However, deactivating and deleting a barclamp that already had been deployed does *not* remove packages installed when the barclamp was deployed. Nor does it stop any services that were started during the barclamp deployment. On the affected node, proceed as follows to undo the deployment:

1. Stop the respective services:

```
root # systemctl stop service
```

2. Disable the respective services:

```
root # systemctl disable service
```

Uninstalling the packages should not be necessary.

10.3.4 Queuing/Dequeuing Proposals

When a proposal is applied to one or more nodes that are not yet available for deployment (for example, because they are rebooting or have not been yet fully installed), the proposal will be put in a queue. A message like

```
Successfully queued the proposal until the following become ready: d52-54-00-6c-25-44
```

will be shown when having applied the proposal. A new button *Dequeue* will also become available. Use it to cancel the deployment of the proposal by removing it from the queue.

11 Installing the OpenStack Nodes

The OpenStack nodes represent the actual cloud infrastructure. Node installation and service deployment is done automatically from the Administration Server. Before deploying the OpenStack services, SUSE Linux Enterprise Server 12 SP4 will be installed on all Control Nodes and Storage Nodes.

To prepare the installation, each node needs to be booted using PXE, which is provided by the `tftp` server from the Administration Server. Afterward you can allocate the nodes and trigger the operating system installation.

11.1 Preparations

Meaningful Node Names

Make a note of the MAC address and the purpose of each node (for example, controller, block storage, object storage, compute). This will make deploying the OpenStack components a lot easier and less error-prone. It also enables you to assign meaningful names (aliases) to the nodes, which are otherwise listed with the MAC address by default.

BIOS Boot Settings

Make sure booting using PXE (booting from the network) is enabled and configured as the *primary* boot-option for each node. The nodes will boot twice from the network during the allocation and installation phase. Booting from the first hard disk needs to be configured as the second boot option.

Custom Node Configuration

All nodes are installed using AutoYaST with the same configuration located at `/opt/dell/chef/cookbooks/provisioner/templates/default/autoyast.xml.erb`. If this configuration does not match your needs (for example if you need special third party drivers) you need to make adjustments to this file. See the <https://documentation.suse.com/sles/15-SP1/single-html/SLES-autoyast/#book-autoyast> for details. If you change the AutoYaST configuration file, you need to re-upload it to Chef using the following command:

```
knife cookbook upload -o /opt/dell/chef/cookbooks/ provisioner
```

Direct `root` Login

By default, the `root` account on the nodes has no password assigned, so a direct `root` login is not possible. Logging in on the nodes as `root` is only possible via SSH public keys (for example, from the Administration Server).

If you want to allow direct `root` login, you can set a password via the Crowbar Provisioner barclamp before deploying the nodes. That password will be used for the `root` account on all OpenStack nodes. Using this method after the nodes are deployed is not possible. In that case you would need to log in to each node via SSH from the Administration Server and change the password manually with `passwd`.

SETTING A `root` PASSWORD FOR THE OPENSTACK NODES

1. Create an md5-hashed `root`-password, for example by using `openssl passwd -1`.
2. Open a browser and point it to the Crowbar Web interface on the Administration Server, for example `http://192.168.124.10`. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it during the installation.
3. Open the barclamp menu by clicking *Barclamps* > *Crowbar*. Click the *Provisioner* barclamp entry and *Edit* the *Default* proposal.
4. Click *Raw* in the *Attributes* section to edit the configuration file.
5. Add the following line to the end of the file before the last closing curly bracket:

```
, "root_password_hash": "HASHED_PASSWORD"
```

replacing `"HASHED_PASSWORD"` with the password you generated in the first step.

6. Click *Apply*.

11.2 Node Installation

To install a node, you need to boot it first using PXE. It will be booted with an image that enables the Administration Server to discover the node and make it available for installation. When you have allocated the node, it will boot using PXE again and the automatic installation will start.

1. Boot all nodes that you want to deploy using PXE. The nodes will boot into the SLEShammer image, which performs the initial hardware discovery.

! Important: Limit the Number of Concurrent Boots using PXE

Booting many nodes at the same time using PXE will cause heavy load on the TFTP server, because all nodes will request the boot image at the same time. We recommend booting the nodes at different intervals.

2. Open a browser and point it to the Crowbar Web interface on the Administration Server, for example `http://192.168.124.10/`. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it.

Click *Nodes* > *Dashboard* to open the *Node Dashboard*.

3. Each node that has successfully booted will be listed as being in state `Discovered`, indicated by a yellow bullet. The nodes will be listed with their MAC address as a name. Wait until all nodes are listed as `Discovered` before proceeding. If a node does not report as `Discovered`, it may need to be rebooted manually.

The screenshot shows the SUSE OpenStack Cloud Crowbar Node Dashboard. At the top, there is a navigation bar with the following items: SUSE® OpenStack Cloud Crowbar, Nodes (selected), Network, Barclamps, Utilities, and Help. Below the navigation bar, the main heading is "Dashboard" with a sub-label "3 nodes". To the right of the heading is a "New group" input field and an "Add" button. A light blue informational banner states: "You may regroup nodes by dragging a node into the desired group. You may drop a node [here] to reset to automatic grouping." Below this banner is a list of nodes under the group "sw-unknown". The nodes are listed as follows:

Node Name	Status
admin	Discovered (green dot)
d52-54-00-19-83-45	Discovered (yellow dot)
d52-54-00-27-4d-a4	Discovered (yellow dot)
d52-54-00-52-9d-47	Discovered (yellow dot)
d52-54-00-52-fa-1c	Discovered (yellow dot)
d52-54-00-69-f8-25	Discovered (yellow dot)
d52-54-00-6a-44-34	Discovered (yellow dot)
d52-54-00-8e-96-71	Discovered (yellow dot)
d52-54-00-af-fa-e7	Discovered (yellow dot)

At the bottom of the dashboard, there is a footer that reads "SUSE® OpenStack Cloud 9 - Provided by SUSE®".

FIGURE 11.1: DISCOVERED NODES

4. Although this step is optional, we recommend properly grouping your nodes at this stage, since it lets you clearly arrange all nodes. Grouping the nodes by role would be one option, for example control, compute and object storage (swift).
 - a. Enter the name of a new group into the *New Group* text box and click *Add Group*.
 - b. Drag and drop a node onto the title of the newly created group. Repeat this step for each node you want to put into the group.

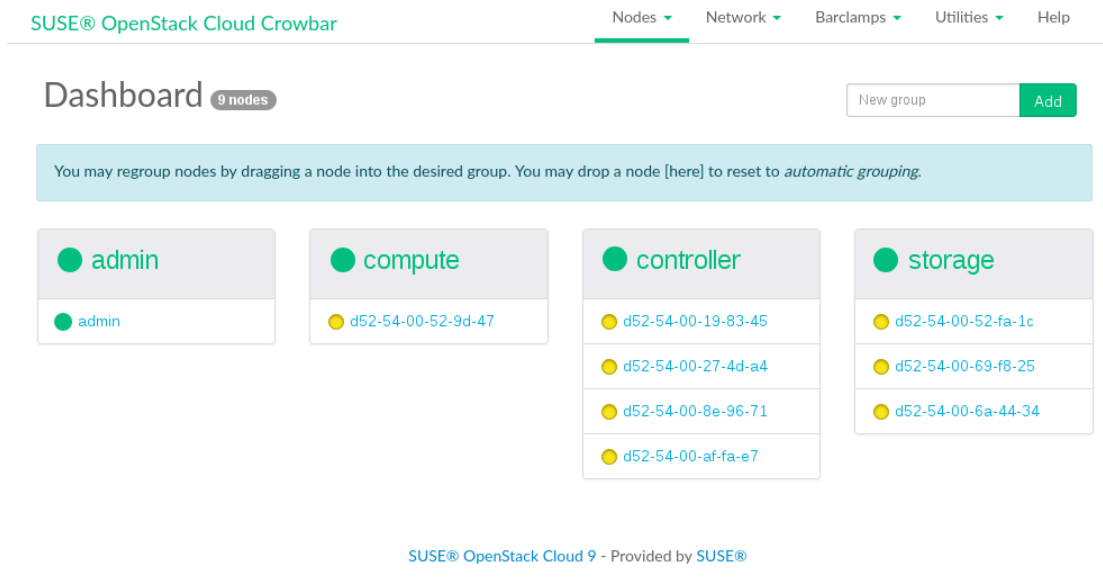


FIGURE 11.2: GROUPING NODES

5. To allocate all nodes, click *Nodes > Bulk Edit*. To allocate a single node, click the name of a node, then click *Edit*.

Edit node

d52-54-77-77-01-01.virtual.cloud.suse.de

Target Platform

SLES 12 SP4

Software RAID

Single disk ▾

Alias

d52-54-77-77-01-01

Public Name

The public name is the hostname that users will use to access services on this node. Re-apply proposals on the node to have this setting taken into account immediately, or wait for an automatic update on the node. Any name specified here should already exist in the upstream DNS zones.

Description**Group**

controller

Intended Role

Controller ▾

Intended Role is the way you intend to use the node in your cloud infrastructure. The value is used to propose the initial node deployment for barclamps.

Filesystem Type

ext4 ▾

Availability Zone

Availability zones allow to arrange sets of either OpenStack Compute or OpenStack Block Storage hosts into logical groups. If empty, the default availability zone will be used.

Cancel

Save

FIGURE 11.3: EDITING A SINGLE NODE

Important: Limit the Number of Concurrent Node Deployments

Deploying many nodes in bulk mode will cause heavy load on the Administration Server. The subsequent concurrent Chef client runs triggered by the nodes will require a lot of RAM on the Administration Server.

Therefore it is recommended to limit the number of concurrent “Allocations” in bulk mode. The maximum number depends on the amount of RAM on the Administration Server—limiting concurrent deployments to five up to ten is recommended.

6. In single node editing mode, you can also specify the *Filesystem Type* for the node. By default, it is set to `ext4` for all nodes. We recommended using the default.
7. Provide a meaningful *Alias*, *Public Name*, and a *Description* for each node, and then check the *Allocate* box. You can also specify the *Intended Role* for the node. This optional setting is used to make reasonable proposals for the barclamps.
By default the *Target Platform* is set to *SLES 12 SP2*.

Tip: Alias Names

Providing an alias name will change the default node names (MAC address) to the name you provided, making it easier to identify the node. Furthermore, this alias will also be used as a DNS `CNAME` for the node in the admin network. As a result, you can access the node via this alias when, for example, logging in via SSH.

Tip: Public Names

A node's *Alias Name* is resolved by the DNS server installed on the Administration Server and therefore only available within the cloud network. The OpenStack Dashboard or some APIs (`keystone-server`, `glance-server`, `cinder-controller`, `neutron-server`, `nova-controller`, and `swift-proxy`) can be accessed from outside the SUSE OpenStack Cloud network. To be able to access them by name, these names need to be resolved by a name server placed outside of the SUSE OpenStack Cloud network. If you have created DNS entries for nodes, specify the name in the *Public Name* field.

The *Public Name* is never used within the SUSE OpenStack Cloud network. However, if you create an SSL certificate for a node that has a public name, this name must be added as an AlternativeName to the certificate. See [Section 2.3, “SSL Encryption”](#) for more information.

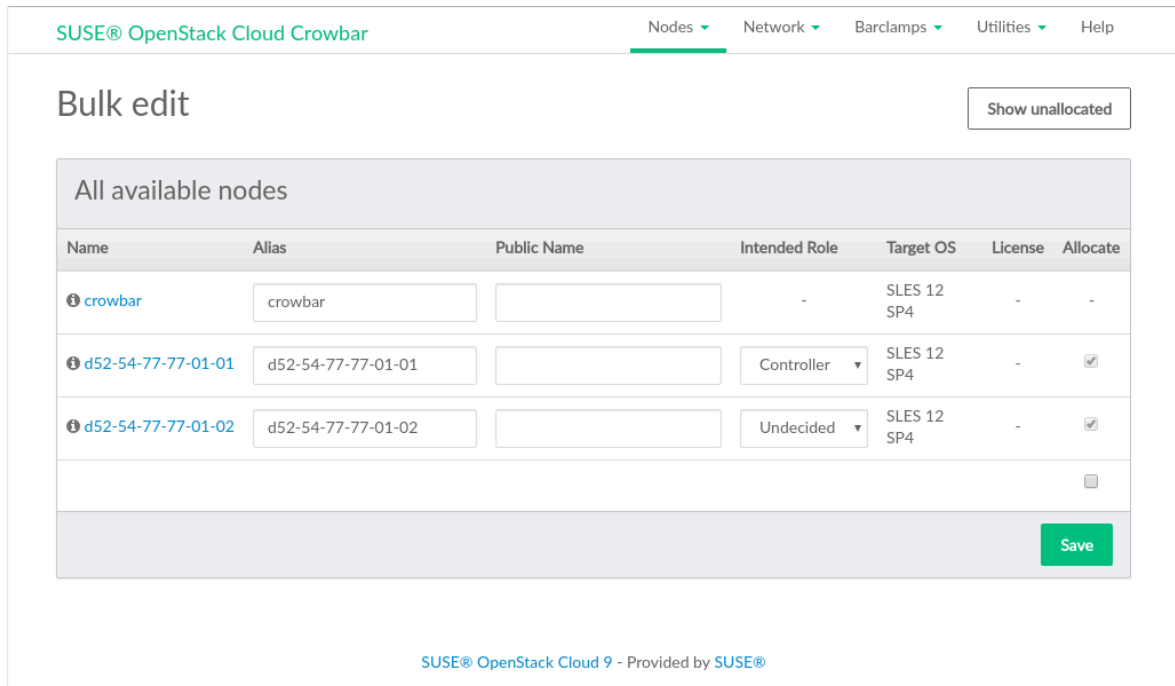


FIGURE 11.4: BULK EDITING NODES

8. When you have filled in the data for all nodes, click *Save*. The nodes will reboot and commence the AutoYaST-based SUSE Linux Enterprise Server installation (or installation of other target platforms, if selected) via a second boot using PXE. Click *Nodes > Dashboard* to return to the *Node Dashboard*.
9. Nodes that are being installed are listed with the status Installing (yellow/green bullet). When the installation of a node has finished, it is listed as being Ready, indicated by a green bullet. Wait until all nodes are listed as Ready before proceeding.

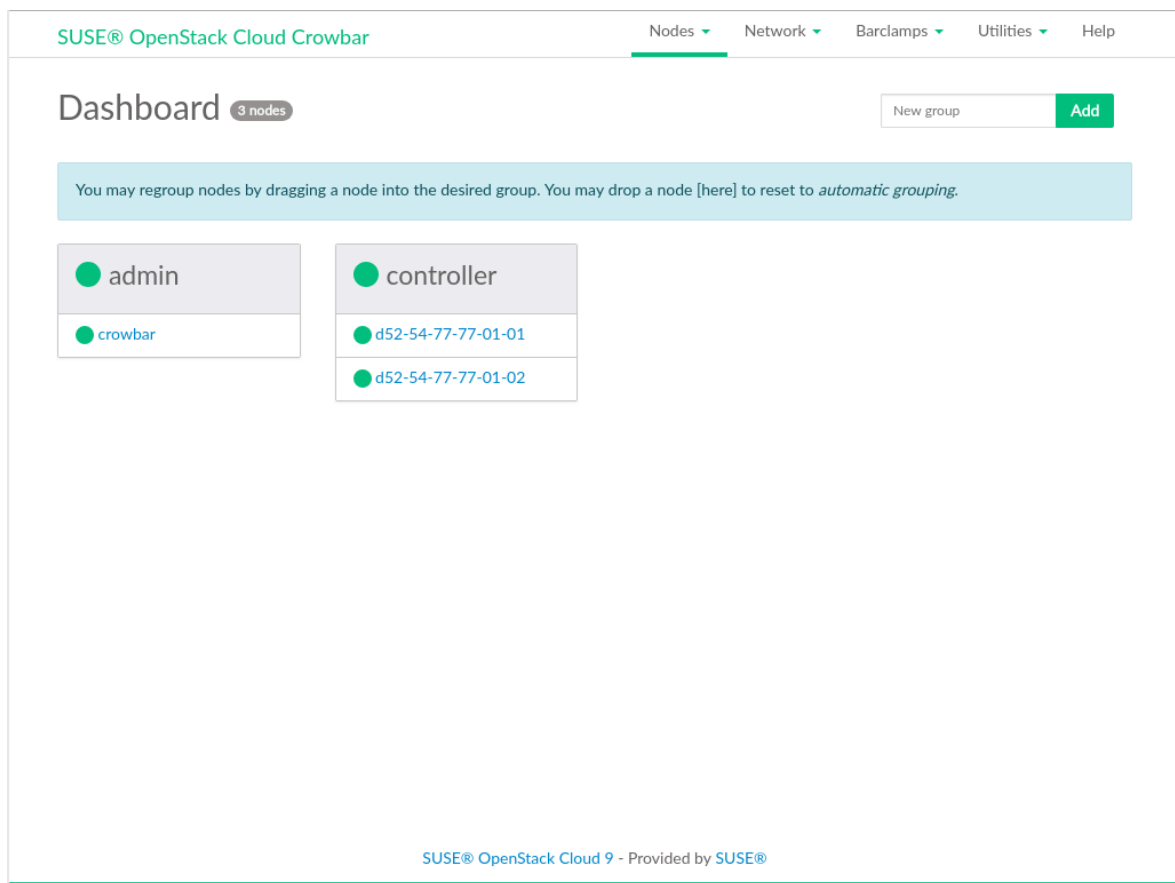


FIGURE 11.5: ALL NODES HAVE BEEN INSTALLED

11.3 Converting Existing SUSE Linux Enterprise Server 12 SP4 Machines Into SUSE OpenStack Cloud Nodes

SUSE OpenStack Cloud allows adding existing machines installed with SUSE Linux Enterprise Server 12 SP4 to the pool of nodes. This enables you to use spare machines for SUSE OpenStack Cloud, and offers an alternative way of provisioning and installing nodes (via SUSE Manager for example). The machine must run SUSE Linux Enterprise Server 12 SP4.

The machine also needs to be on the same network as the Administration Server, because it needs to communicate with this server. Since the Administration Server provides a DHCP server, we recommend configuring this machine to get its network assignments from DHCP. If it has a static IP address, make sure it is not already used in the admin network. Check the list of used IP addresses with the YaST Crowbar module as described in [Section 7.2, “Networks”](#).

Proceed as follows to convert an existing SUSE Linux Enterprise Server 12 SP4 machine into a SUSE OpenStack Cloud node:

1. Download the `crowbar_register` script from the Administration Server at http://192.168.124.10:8091/suse-12.4/x86_64/crowbar_register. Replace the IP address with the IP address of your Administration Server using `curl` or `wget`. Note that the download only works from within the admin network.
2. Make the `crowbar_register` script executable (`chmod a+x crowbar_register`).
3. Run the `crowbar_register` script. If you have multiple network interfaces, the script tries to automatically detect the one that is connected to the admin network. You may also explicitly specify which network interface to use by using the `--interface` switch, for example `crowbar_register --interface eth1`.
4. After the script has successfully run, the machine has been added to the pool of nodes in the SUSE OpenStack Cloud and can be used as any other node from the pool.

11.4 Post-Installation Configuration

The following lists some *optional* configuration steps like configuring node updates, monitoring, access, and enabling SSL. You may entirely skip the following steps or perform any of them at a later stage.

11.4.1 Deploying Node Updates with the Updater Barclamp

To keep the operating system and the SUSE OpenStack Cloud software itself up-to-date on the nodes, you can deploy either the Updater barclamp or the SUSE Manager barclamp. The latter requires access to a SUSE Manager server. The Updater barclamp uses Zypper to install updates and patches from repositories made available on the Administration Server.

The easiest way to provide the required repositories on the Administration Server is to set up an SMT server as described in [Chapter 4, Installing and Setting Up an SMT Server on the Administration Server \(Optional\)](#). Alternatives to setting up an SMT server are described in [Chapter 5, Software Repository Setup](#).

The Updater barclamp lets you deploy updates that are available on the update repositories at the moment of deployment. Each time you deploy updates with this barclamp you can choose a different set of nodes to which the updates are deployed. This lets you exactly control where and when updates are deployed.

To deploy the Updater barclamp, proceed as follows. For general instructions on how to edit barclamp proposals refer to [Section 10.3, “Deploying Barclamp Proposals”](#).

1. Open a browser and point it to the Crowbar Web interface on the Administration Server, for example `http://192.168.124.10/`. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it during the installation.
2. Open the barclamp menu by clicking *Barclamps > Crowbar*. Click the *Updater* barclamp entry and *Create* to open the proposal.
3. Configure the barclamp by the following attributes. This configuration always applies to all nodes on which the barclamp is deployed. Individual configurations for certain nodes are only supported by creating a separate proposal.

Use zypper

Define which Zypper subcommand to use for updating. *patch* will install all patches applying to the system from the configured update repositories that are available. *update* will update packages from all configured repositories (not just the update repositories) that have a higher version number than the installed packages. *dist-upgrade* replaces each package installed with the version from the repository and deletes packages not available in the repositories.

We recommend using *patch*.

Enable GPG Checks

If set to true (recommended), checks if packages are correctly signed.

Automatically Agree With Licenses

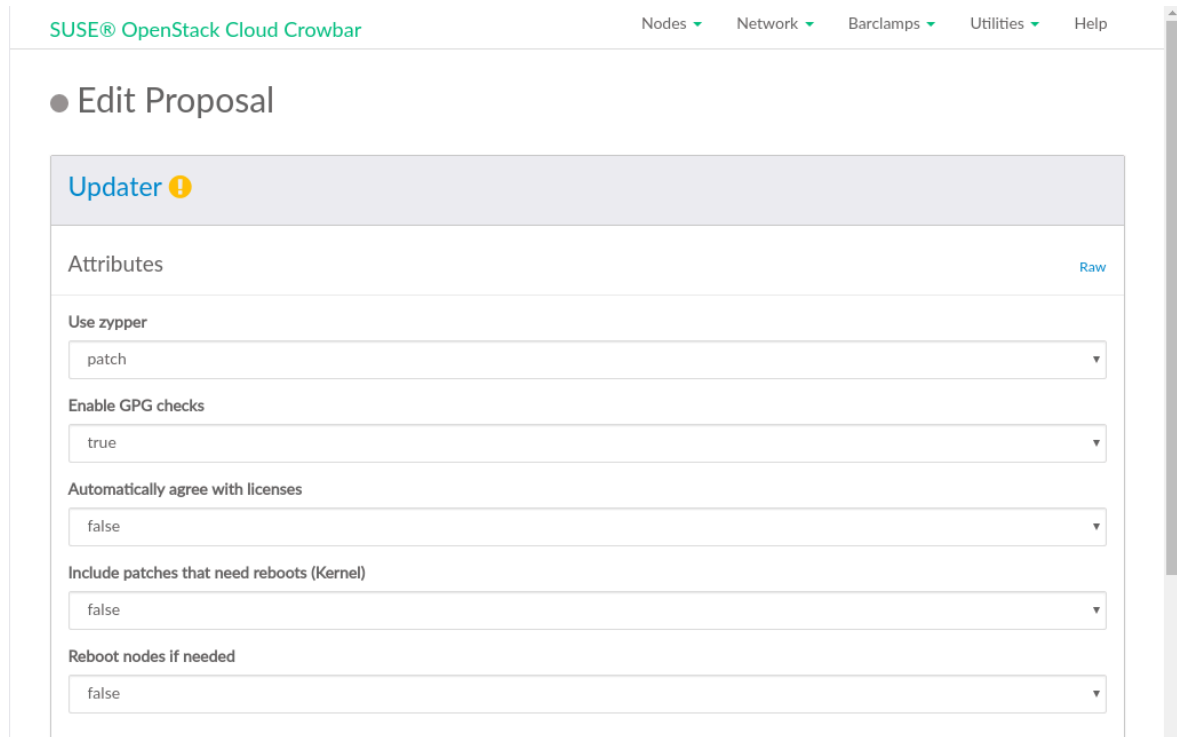
If set to true (recommended), Zypper automatically accepts third party licenses.

Include Patches that need Reboots (Kernel)

Installs patches that require a reboot (for example Kernel or glibc updates). Only set this option to `true` when you can safely reboot the affected nodes. Refer to *Book “Operations Guide Crowbar”, Chapter 1 “Maintenance”, Section 1.1 “Keeping the Nodes Up-To-Date”* for more information. Installing a new Kernel and not rebooting may result in an unstable system.

Reboot Nodes if Needed

Automatically reboots the system in case a patch requiring a reboot has been installed. Only set this option to `true` when you can safely reboot the affected nodes. Refer to *Book “Operations Guide Crowbar”, Chapter 1 “Maintenance”, Section 1.1 “Keeping the Nodes Up-To-Date”* for more information.



The screenshot shows the 'Edit Proposal' page for the 'Updater' barclamp in the SUSE OpenStack Cloud Crowbar interface. The page has a navigation bar at the top with 'Nodes', 'Network', 'Barclamps', 'Utilities', and 'Help' menus. The main heading is 'Edit Proposal'. Below this, there is a section for the 'Updater' barclamp, which includes a 'Raw' link and several configuration options:

- Use zypper:** A dropdown menu with 'patch' selected.
- Enable GPG checks:** A dropdown menu with 'true' selected.
- Automatically agree with licenses:** A dropdown menu with 'false' selected.
- Include patches that need reboots (Kernel):** A dropdown menu with 'false' selected.
- Reboot nodes if needed:** A dropdown menu with 'false' selected.

FIGURE 11.6: SUSE UPDATER BARCLAMP: CONFIGURATION

4. Choose the nodes on which the Updater barclamp should be deployed in the *Node Deployment* section by dragging them to the *Updater* column.

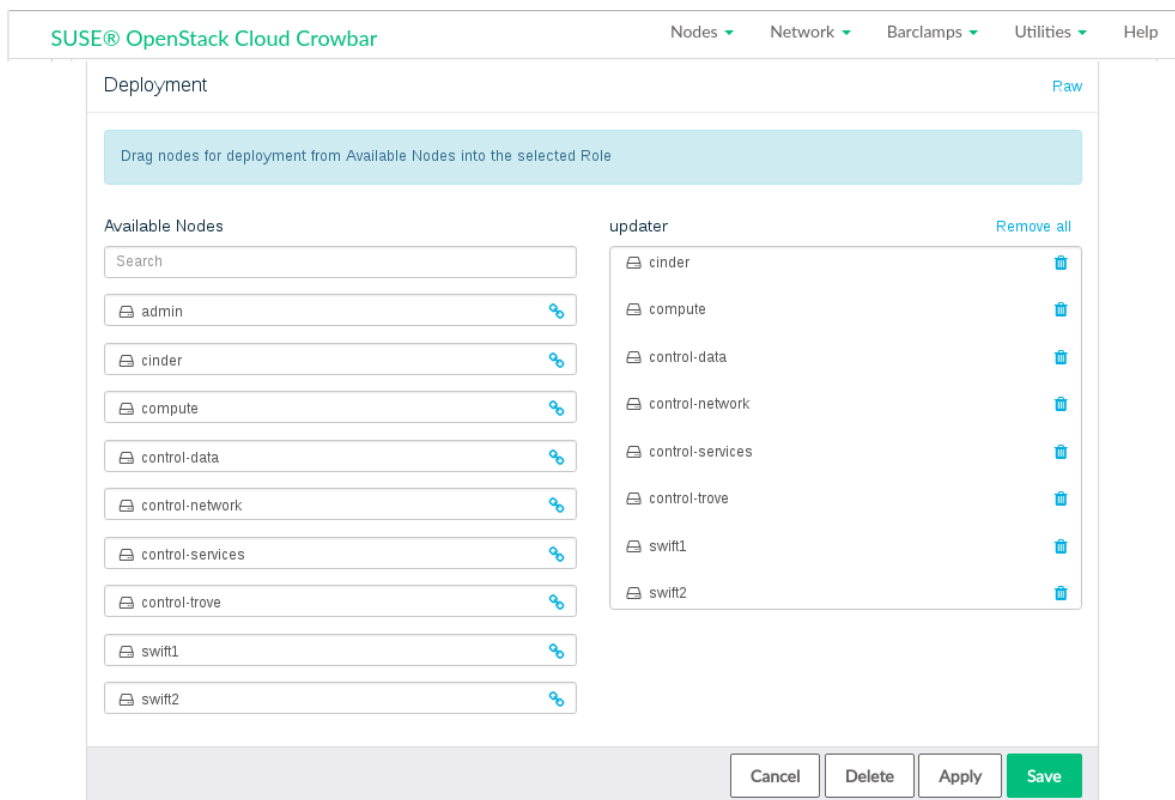


FIGURE 11.7: SUSE UPDATER BARCLAMP: NODE DEPLOYMENT

zypper keeps track of the packages and patches it installs in `/var/log/zypp/history`. Review that log file on a node to find out which updates have been installed. A second log file recording debug information on the **zypper** runs can be found at `/var/log/zypper.log` on each node.



Warning: Updating Software Packages on Cluster Nodes

Before starting an update for a cluster node, either stop the cluster stack on that node or put the cluster into maintenance mode. If the cluster resource manager on a node is active during the software update, this can lead to unpredictable results like fencing of active nodes. For detailed instructions refer to <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#sec-ha-clvm-migrate>.

11.4.2 Configuring Node Updates with the *SUSE Manager Client* Barclamp

To keep the operating system and the SUSE OpenStack Cloud software itself up-to-date on the nodes, you can deploy either *SUSE Manager Client* barclamp or the Updater barclamp. The latter uses Zypper to install updates and patches from repositories made available on the Administration Server.

To enable the SUSE Manager server to manage the SUSE OpenStack Cloud nodes, you must make the respective SUSE OpenStack Cloud Crowbar 9 channels, the SUSE Linux Enterprise Server 12 SP4 channels, and the channels for extensions used with your deployment (High Availability Extension, SUSE Enterprise Storage) available via an activation key.

The *SUSE Manager Client* barclamp requires access to the SUSE Manager server from every node it is deployed to.

To deploy the *SUSE Manager Client* barclamp, proceed as follows. For general instructions on how to edit barclamp proposals refer to [Section 10.3, “Deploying Barclamp Proposals”](#).

1. Download the package `rhncert-ssl-cert-VERSION-RELEASE.noarch.rpm` from `https://susemanager.example.com/pub/`. `VERSION` and `RELEASE` may vary, ask the administrator of the SUSE Manager for the correct values. `susemanager.example.com` needs to be replaced by the address of your SUSE Manager server. Copy the file you downloaded to `/opt/dell/chef/cookbooks/suse-manager-client/files/default/ssl-cert.rpm` on the Administration Server. The package contains the SUSE Manager's CA SSL Public Certificate. The certificate installation has not been automated on purpose, because downloading the certificate manually enables you to check it before copying it.
2. Re-install the barclamp by running the following command:

```
/opt/dell/bin/barclamp_install.rb --rpm core
```
3. Open a browser and point it to the Crowbar Web interface on the Administration Server, for example `http://192.168.124.10/`. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it during the installation.
4. Open the barclamp menu by clicking *Barclamps* > *Crowbar*. Click the *SUSE Manager Client* barclamp entry and *Create* to open the proposal.
5. Specify the URL of the script for activation of the clients in the *URL of the bootstrap script* field.

6. Choose the nodes on which the SUSE Manager barclamp should be deployed in the *Deployment* section by dragging them to the *suse-manager-client* column. We recommend deploying it on all nodes in the SUSE OpenStack Cloud.

The screenshot shows the 'Edit Proposal' interface for the 'SUSE Manager Client' barclamp. The interface is titled 'SUSE Manager Client' with a warning icon. It is divided into three main sections: 'Attributes', 'Deployment', and 'Available Nodes'. The 'Attributes' section contains a text box with manual steps for installation and a field for the 'URL of the bootstrap script'. The 'Deployment' section has a 'Drag nodes for deployment from Available Nodes into the selected Role' instruction and a list of available nodes: 'crowbar', 'd52-54-77-77-01-01', and 'd52-54-77-77-01-02'. The 'Available Nodes' section shows a search bar and a list of nodes with drag handles. The 'suse-manager-client' role is selected, and there is a 'Remove all' button. At the bottom, there are 'Cancel', 'Delete', 'Apply', and 'Save' buttons. The footer indicates 'SUSE® OpenStack Cloud 9 - Provided by SUSE®'.

FIGURE 11.8: SUSE MANAGER BARCLAMP



Warning: Updating Software Packages on Cluster Nodes

Before starting an update for a cluster node, either stop the cluster stack on that node or put the cluster into maintenance mode. If the cluster resource manager on a node is active during the software update, this can lead to unpredictable results like fencing of active nodes. For detailed instructions refer to <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#sec-ha-clvm-migrate>.

11.4.3 Mounting NFS Shares on a Node

The NFS barclamp allows you to mount NFS share from a remote host on nodes in the cloud. This feature can, for example, be used to provide an image repository for glance. Note that all nodes which are to mount an NFS share must be able to reach the NFS server. This requires manually adjusting the network configuration.

To deploy the NFS barclamp, proceed as follows. For general instructions on how to edit barclamp proposals refer to [Section 10.3, “Deploying Barclamp Proposals”](#).

1. Open a browser and point it to the Crowbar Web interface on the Administration Server, for example <http://192.168.124.10/>. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it during the installation.
2. Open the barclamp menu by clicking *Barclamps* > *Crowbar*. Click the *NFS Client* barclamp entry and *Create* to open the proposal.
3. Configure the barclamp by the following attributes. Each set of attributes is used to mount a single NFS share.

Name

Unique name for the current configuration. This name is used in the Web interface only to distinguish between different shares.

NFS Server

Fully qualified host name or IP address of the NFS server.

Export

Export name for the share on the NFS server.

Path

Mount point on the target machine.

Mount Options

Mount options that will be used on the node. See [man 8 mount](#) for general mount options and [man 5 nfs](#) for a list of NFS-specific options. Note that the general option [nofail](#) (do not report errors if device does not exist) is automatically set.

4. After having filled in all attributes, click *Add*. If you want to mount more than one share, fill in the data for another NFS mount. Otherwise click *Save* to save the data, or *Apply* to deploy the proposal. Note that you must always click *Add* before saving or applying the barclamp, otherwise the data that was entered will be lost.

NFS Client: Proposal 1 ⓘ

Attributes Raw

Name	NFS Server	Export	Path	Mount options
Currently there are no mounts available				

Name NFS Server Export Path Mount options

The "nofail" option will automatically be added to the options

FIGURE 11.9: NFS BARCLAMP

5. Go to the *Node Deployment* section and drag and drop all nodes, on which the NFS shares defined above should be mounted, to the *nfs-client* column. Click *Apply* to deploy the proposal.

The NFS barclamp is the only barclamp that lets you create different proposals, enabling you to mount different NFS shares on different nodes. When you have created an NFS proposal, a special *Edit* is shown in the barclamp overview of the Crowbar Web interface. Click it to either *Edit* an existing proposal or *Create* a new one. New proposals must have unique names.

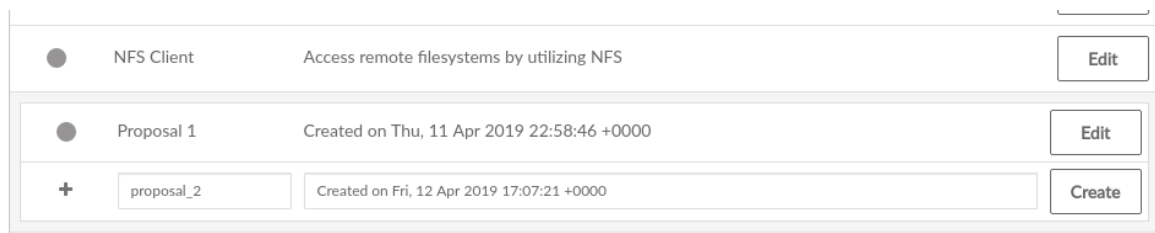


FIGURE 11.10: EDITING AN NFS BARCLAMP PROPOSAL

11.4.4 Using an Externally Managed Ceph Cluster

The following chapter provides instructions on using an external Ceph cluster in SUSE OpenStack Cloud Crowbar.

11.4.4.1 Requirements

Ceph Release

External Ceph cluster are supported with SUSE Enterprise Storage 5 or higher. The version of Ceph should be compatible with the version of the Ceph client supplied with SUSE Linux Enterprise Server 12 SP4.

Network Configuration

The external Ceph cluster needs to be connected to a separate VLAN, which is mapped to the SUSE OpenStack Cloud storage VLAN. See [Section 2.1, "Network"](#) for more information.

11.4.4.2 Making Ceph Available on the SUSE OpenStack Cloud Nodes

Ceph can be used from the KVM Compute Nodes, with cinder, and with glance. The following installation steps need to be executed on each node accessing Ceph:

Important: Installation Workflow

The following steps need to be executed before the barclamps get deployed.

1. Log in as user `root` to a machine in the Ceph cluster and generate keyring files for cinder users. Optionally, you can generate keyring files for the glance users (only needed when using glance with Ceph/Rados). The keyring file that will be generated for cinder will also be used on the Compute Nodes. To do so, you need to specify pool names and user names for both services. The default names are:

	glance	cinder
User	glance	cinder
Pool	images	volumes

Make a note of user and pool names in case you do not use the default values. You will need this information later, when deploying glance and cinder.

2.  **Warning: Automatic Changes to the Cluster**

If you decide to use the admin keyring file to connect the external Ceph cluster, be aware that after Crowbar discovers this admin keyring, it will create client keyring files, pools, and capabilities needed to run glance, cinder, or nova integration.

If you have access to the admin keyring file and agree that automatic changes will be done to the cluster as described above, copy it together with the Ceph configuration file to the Administration Server. If you cannot access this file, create a keyring:

- a. When you can access the admin keyring file `ceph.client.admin.keyring`, copy it together with `ceph.conf` (both files are usually located in `/etc/ceph`) to a temporary location on the Administration Server, for example `/root/tmp/`.
- b. If you cannot access the admin keyring file create a new keyring file with the following commands. Re-run the commands for glance, too, if needed. First create a key:

```
ceph auth get-or-create-key client.USERNAME mon "allow r" \  
osd 'allow class-read object_prefix rbd_children, allow rwx \  
pool=POOLNAME'
```

Replace `USERNAME` and `POOLNAME` with the respective values.

Now use the key to generate the keyring file `/etc/ceph/ceph.client.USER-NAME.keyring`:

```
ceph-authtool \  
/etc/ceph/ceph.client.USERNAME.keyring \  
--create-keyring --name=client.USERNAME> \  
--add-key=KEY
```

Replace `USERNAME` with the respective value.

Copy the Ceph configuration file `ceph.conf` (usually located in `/etc/ceph`) and the keyring file(s) generated above to a temporary location on the Administration Server, for example `/root/tmp/`.

3. Log in to the Crowbar Web interface and check whether the nodes which should have access to the Ceph cluster already have an IP address from the storage network. Do so by going to the *Dashboard* and clicking the node name. An *IP address* should be listed for *storage*. Make a note of the *Full name* of each node that has *no* storage network IP address.
4. Log in to the Administration Server as user `root` and run the following command for all nodes you noted down in the previous step:

```
crowbar network allocate_ip "default" NODE "storage" "host"  
chef-client
```

`NODE` needs to be replaced by the node's name.

5. After executing the command in the previous step for all affected nodes, run the command `chef-client` on the Administration Server.
6. Log in to each affected node as user `root`. See [Q:](#) for instructions. On each node, do the following:
 - a. Manually install `nova`, `cinder` (if using `cinder`) and/or `glance` (if using `glance`) packages with the following commands:

```
zypper in openstack-glance  
zypper in openstack-cinder  
zypper in openstack-nova
```

- b. Copy the `ceph.conf` file from the Administration Server to `/etc/ceph`:

```
mkdir -p /etc/ceph  
scp root@admin:/root/tmp/ceph.conf /etc/ceph
```

```
chmod 664 /etc/ceph/ceph.conf
```

c. Copy the keyring file(s) to `/etc/ceph`. The exact process depends on whether you have copied the admin keyring file or whether you have created your own keyrings:

- i. If you have copied the admin keyring file, run the following command on the Control Node(s) on which cinder and glance will be deployed, and on all KVM Compute Nodes:

```
scp root@admin:/root/tmp/ceph.client.admin.keyring /etc/ceph
chmod 640 /etc/ceph/ceph.client.admin.keyring
```

- ii. If you have created your own keyrings, run the following command on the Control Node on which cinder will be deployed, and on all KVM Compute Nodes to copy the cinder keyring:

```
scp root@admin:/root/tmp/ceph.client.cinder.keyring /etc/ceph
chmod 640 /etc/ceph/ceph.client.cinder.keyring
```

On Control Node on which cinder will be deployed run the following command to update file ownership:

```
chown root.cinder /etc/ceph/ceph.client.cinder.keyring
```

On KVM Compute Nodes run the following command to update file ownership:

```
chown root.nova /etc/ceph/ceph.client.cinder.keyring
```

Now copy the glance keyring to the Control Node on which glance will be deployed:

```
scp root@admin:/root/tmp/ceph.client.glance.keyring /etc/ceph
chmod 640 /etc/ceph/ceph.client.glance.keyring
chown root.glance /etc/ceph/ceph.client.glance.keyring
```

11.4.5 Accessing the Nodes

The nodes can only be accessed via SSH from the Administration Server—it is not possible to connect to them from any other host in the network.

The `root` account *on the nodes* has no password assigned, therefore logging in to a node as `root@node` is only possible via SSH with key authentication. By default, you can only log in with the key of the `root` of the Administration Server (`root@admin`) via SSH only.

If you have added users to the Administration Server and want to give them permission to log in to the nodes as well, you need to add these users' public SSH keys to `root`'s `authorized_keys` file on all nodes. Proceed as follows:

PROCEDURE 11.1: COPYING SSH KEYS TO ALL NODES

1. If they do not already exist, generate an SSH key pair with `ssh-keygen`. This key pair belongs to the user that you use to log in to the nodes. Alternatively, copy an existing public key with `ssh-copy-id`. Refer to the respective man pages for more information.
2. Log in to the Crowbar Web interface on the Administration Server, for example `http://192.168.124.10/` (user name and default password: `crowbar`).
3. Open the barclamp menu by clicking *Barclamps* > *Crowbar*. Click the *Provisioner* barclamp entry and *Edit* the *Default* proposal.
4. Copy and paste the *public* SSH key of the user into the *Additional SSH Keys* text box. If adding keys for multiple users, note that each key needs to be placed on a new line.
5. Click *Apply* to deploy the keys and save your changes to the proposal.

11.4.6 Enabling SSL

To enable SSL to encrypt communication within the cloud (see [Section 2.3, "SSL Encryption"](#) for details), all nodes running encrypted services need SSL certificates. An SSL certificate is, at a minimum, required on the Control Node.

Each certificate consists of a pair of files: the certificate file (for example, `signing_cert.pem`) and the key file (for example, `signing_key.pem`). If you use your own certificate authority (CA) for signing, you will also need a certificate file for the CA (for example, `ca.pem`). We recommend copying the files to the `/etc` directory using the directory structure outlined below. If you use a dedicated certificate for each service, create directories named after the services (for example, `/etc/keystone`). If you are using shared certificates, use a directory such as `/etc/cloud`.

RECOMMENDED LOCATIONS FOR SHARED CERTIFICATES

SSL Certificate File

`/etc/cloud/ssl/certs/signing_cert.pem`

SSL Key File

/etc/cloud/private/signing_key.pem

CA Certificates File

/etc/cloud/ssl/certs/ca.pem

11.5 Editing Allocated Nodes

All nodes that have been allocated can be decommissioned or re-installed. Click a node's name in the *Node Dashboard* to open a screen with the node details. The following options are available:

Forget

Deletes a node from the pool. If you want to re-use this node again, it needs to be reallocated and re-installed from scratch.

Reinstall

Triggers a reinstallation. The machine stays allocated. Any barclamps that were deployed on the machine will be re-applied after the installation.

Deallocate

Temporarily removes the node from the pool of nodes. After you reallocate the node it will take its former role. Useful for adding additional machines in times of high load or for decommissioning machines in times of low load.

Power Actions > Reboot

Reboots the node.

Power Actions > Shutdown

Shuts the node down.

Power Actions > Power Cycle

Forces a (non-clean) shuts down and a restart afterward. Only use if a reboot does not work.

Power Actions > Power Off

Forces a (non-clean) node shut down. Only use if a clean shut down does not work.

Node

● d52-54-77-77-01-02
Edit

Full Name	d52-54-77-77-01-02.virtual.cloud.suse.de	Hardware	Standard PC (i440FX + PIIX, 1996)
Public Name	—	Service Tag	Not Specified (Not Specified)
Description	—	CPU	Intel Core Processor (Haswell, no TSX) (x86_64)
Target Platform	SLES 12 SP4	Memory	2.43 GB
Uptime	2 days 22 hours 03 minutes 15 seconds	Disk Drives	2
Allocated	Allocated	MAC Address	52:54:77:77:01:02
State	Ready	Switch Name/Port	Unknown / Unknown
Intended Role	Undecided		
Availability Zone	—		
IP Address	<ul style="list-style-type: none"> • admin <ul style="list-style-type: none"> ◦ eth0: 192.168.124.81 		
Links	<ul style="list-style-type: none"> • No links available 		
Applied Barclamps	<ul style="list-style-type: none"> • Crowbar <ul style="list-style-type: none"> ◦ DNS ◦ Deployer ◦ IPMI ◦ Logging ◦ NTP ◦ Network ◦ Provisioner 		
Applied Roles	<ul style="list-style-type: none"> • Crowbar <ul style="list-style-type: none"> ◦ bmc-nat-client ◦ deployer-client ◦ dns-client ◦ ipmi ◦ logging-client ◦ network ◦ ntp-client ◦ provisioner-base 		

Forget
Reinstall
Deallocate

Power Actions ▾

FIGURE 11.11: NODE INFORMATION



Warning: Editing Nodes in a Production System

When de-allocating nodes that provide essential services, the complete cloud will become unusable. If you have not disabled redundancy, you can disable single storage nodes or single compute nodes. However, disabling Control Node(s) will cause major problems.

It will either “kill” certain services (for example swift) or, at worst the complete cloud (when deallocating the Control Node hosting neutron). You should also not disable the nodes providing swift ring and proxy services.

12 Deploying the OpenStack Services

After the nodes are installed and configured you can start deploying the OpenStack components to finalize the installation. The components need to be deployed in a given order, because they depend on one another. The *Pacemaker* component for an HA setup is the only exception from this rule—it can be set up at any time. However, when deploying SUSE OpenStack Cloud Crowbar from scratch, we recommend deploying the *Pacemaker* proposal(s) first. Deployment for all components is done from the Crowbar Web interface through recipes, so-called “barclamps”. (See [Section 12.24, “Roles and Services in SUSE OpenStack Cloud Crowbar”](#) for a table of all roles and services, and how to start and stop them.)

The components controlling the cloud, including storage management and control components, need to be installed on the Control Node(s) (refer to [Section 1.2, “The Control Node\(s\)”](#) for more information). However, you may *not* use your Control Node(s) as a compute node or storage host for swift. Do not install the components *swift-storage* and *nova-compute-** on the Control Node(s). These components must be installed on dedicated Storage Nodes and Compute Nodes. When deploying an HA setup, the Control Nodes are replaced by one or more controller clusters consisting of at least two nodes, and three are recommended. We recommend setting up three separate clusters for data, services, and networking. See [Section 2.6, “High Availability”](#) for more information on requirements and recommendations for an HA setup.

The OpenStack components need to be deployed in the following order. For general instructions on how to edit and deploy barclamps, refer to [Section 10.3, “Deploying Barclamp Proposals”](#). Any optional components that you elect to use must be installed in their correct order.

1. [Deploying designate](#)
2. [Deploying Pacemaker \(Optional, HA Setup Only\)](#)
3. [Deploying the Database](#)
4. [Deploying RabbitMQ](#)
5. [Deploying keystone](#)
6. [Deploying monasca \(Optional\)](#)
7. [Deploying swift \(optional\)](#)
8. [Deploying glance](#)
9. [Deploying cinder](#)

10. *Deploying neutron*
11. *Deploying nova*
12. *Deploying horizon (OpenStack Dashboard)*
13. *Deploying heat (Optional)*
14. *Deploying ceilometer (Optional)*
15. *Deploying manila*
16. *Deploying Tempest (Optional)*
17. *Deploying Magnum (Optional)*
18. *Deploying Octavia*

12.1 Deploying designate

designate provides SUSE OpenStack Cloud Crowbar DNS as a Service (DNSaaS). It is used to create and propagate zones and records over the network using pools of DNS servers. Deployment defaults are in place, so not much is required to configure designate. neutron needs additional settings for integration with designate, which are also present in the `[designate]` section in neutron configuration.

The designate barclamp relies heavily on the DNS barclamp and expects it to be applied without any failures.



Note

In order to deploy designate, at least one node is necessary in the DNS barclamp that is not the admin node. The admin node is not added to the public network. So another node is needed that can be attached to the public network and appear in the designate default pool.

We recommend that DNS services are running in a cluster in highly available deployments where Designate services are running in a cluster. For example, in a typical HA deployment where the controllers are deployed in a 3-node cluster, the DNS barclamp should be applied to all the controllers, in the same manner as Designate.

designate-server role

Installs the designate server packages and configures the mini-dns (mdns) service required by designate.

designate-worker role

Configures a designate worker on the selected nodes. designate uses the workers to distribute its workload.

designate Sink is an optional service and is not configured as part of this barclamp.

designate uses pool(s) over which it can distribute zones and records. Pools can have varied configuration. Any misconfiguration can lead to information leakage.

The designate barclamp creates default Bind9 pool out of the box, which can be modified later as needed. The default Bind9 pool configuration is created by Crowbar on a node with designate-server role in /etc/designate/pools.crowbar.yaml. You can copy this file and edit it according to your requirements. Then provide this configuration to designate using the command:

```
tux > designate-manage pool update --file /etc/designate/pools.crowbar.yaml
```

The dns_domain specified in neutron configuration in [designate] section is the default Zone where DNS records for neutron resources are created via neutron-designate integration. If this is desired, you have to create this zone explicitly using the following command:

```
ardana > openstack zone create < email > < dns_domain >
```

Editing the designate proposal:

● Edit Proposal

Designate ✓ Save Apply Deactivate Cancel

Attributes Custom

```
1 {
2   "debug": false,
3   "use_syslog": true,
4   "keystone_instance": "default",
5   "database_instance": "default",
6   "rabbitmq_instance": "default",
7   "service_user": "designate",
8   "service_password": "",
9   "memcache_secret_key": "",
10  "api": {
11    "protocol": "http",
12    "bind_open_address": true,
13    "bind_port": 9001
14  },
15  "db": {
16    "password": "",
17    "user": "designate",
18    "database": "designate"
19  }
20 }
```

Deployment Raw

Drag nodes for deployment from Available Nodes into the selected Role

Available Nodes

Search

- compute1
- crowbar
- dashboard

designate-server Remove all

- dashboard

designate-worker Remove all

- dashboard

Save Apply Deactivate Cancel

12.1.1 Using PowerDNS Backend

Designate uses Bind9 backend by default. It is also possible to use PowerDNS backend in addition to, or as an alternative, to Bind9 backend. To do so PowerDNS must be manually deployed as The designate barclamp currently does not provide any facility to automatically install and configure PowerDNS. This section outlines the steps to deploy PowerDNS backend.



Note

If PowerDNS is already deployed, you may skip the [Section 12.1.1.1, “Install PowerDNS”](#) section and jump to the [Section 12.1.1.2, “Configure Designate To Use PowerDNS Backend”](#) section.

12.1.1.1 Install PowerDNS

Follow these steps to install and configure PowerDNS on a Crowbar node. Keep in mind that PowerDNS must be deployed with MySQL backend.



Note

We recommend that PowerDNS are running in a cluster in highly availability deployments where Designate services are running in a cluster. For example, in a typical HA deployment where the controllers are deployed in a 3-node cluster, PowerDNS should be running on all the controllers, in the same manner as Designate.

1. Install PowerDNS packages.

```
root # zypper install pdns pdns-backend-mysql
```

2. Edit `/etc/pdns/pdns.conf` and provide these options: (See <https://doc.powerdns.com/authoritative/settings.html> [↗](#) for a complete reference).

api

Set it to yes to enable Web service Rest API.

api-key

Static Rest API access key. Use a secure random string here.

launch

Must set to gmysql to use MySQL backend.

gmysql-host

Hostname (i.e. FQDN) or IP address of the MySQL server.

gmysql-user

MySQL user which have full access to the PowerDNS database.

gmysql-password

Password for the MySQL user.

gmysql-dbname

MySQL database name for PowerDNS.

local-port

Port number where PowerDNS is listening for upcoming requests.

setgid

The group where the PowerDNS process is running under.

setuid

The user where the PowerDNS process is running under.

webserver

Must set to yes to enable web service RestAPI.

webserver-address

Hostname (FQDN) or IP address of the PowerDNS web service.

webserver-allow-from

List of IP addresses (IPv4 or IPv6) of the nodes that are permitted to talk to the PowerDNS web service. These must include the IP address of the Designate worker nodes.

For example:

```
api=yes
api-key=Sfw234sDFw90z
launch=gmysql
gmysql-host=mysql.acme.com
gmysql-user=powerdns
gmysql-password=SuperSecured123
gmysql-dbname=powerdns
local-port=54
setgid=pdns
setuid=pdns
webserver=yes
webserver-address=192.168.124.83
webserver-allow-from=0.0.0.0/0,::/0
```

3. Login to MySQL from a Crowbar MySQL node and create the PowerDNS database and the user which has full access to the PowerDNS database. Remember, the database name, user name, and password must match `gmysql-dbname`, `gmysql-user`, and `gmysql-password` that were specified above respectively.

For example:

```
root # mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 20075
Server version: 10.2.29-MariaDB-log SUSE package

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE powerdns;
Query OK, 1 row affected (0.01 sec)

MariaDB [(none)]> GRANT ALL ON powerdns.* TO 'powerdns'@'localhost' IDENTIFIED BY
'SuperSecured123';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL ON powerdns.* TO 'powerdns'@'192.168.124.83' IDENTIFIED
BY 'SuperSecured123';
Query OK, 0 rows affected, 1 warning (0.02 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)

MariaDB [(none)]> exit
Bye
```

4. Create a MySQL schema file, named `powerdns-schema.sql`, with the following content:

```
/*
SQL statements to create tables in designate_pdns DB.
Note: This file is taken as is from:
https://raw.githubusercontent.com/openstack/designate/master/devstack/
designate\_plugins/backend-pdns4-mysql-db.sql
*/
CREATE TABLE domains (
  id          INT AUTO_INCREMENT,
  name       VARCHAR(255) NOT NULL,
  master     VARCHAR(128) DEFAULT NULL,
  last_check INT DEFAULT NULL,
  type       VARCHAR(6) NOT NULL,
```

```

    notified_serial      INT DEFAULT NULL,
    account              VARCHAR(40) DEFAULT NULL,
    PRIMARY KEY (id)
) Engine=InnoDB;

CREATE UNIQUE INDEX name_index ON domains(name);

CREATE TABLE records (
    id                   INT AUTO_INCREMENT,
    domain_id            INT DEFAULT NULL,
    name                 VARCHAR(255) DEFAULT NULL,
    type                 VARCHAR(10) DEFAULT NULL,
    -- Changed to "TEXT", as VARCHAR(65000) is too big for most MySQL installs
    content              TEXT DEFAULT NULL,
    ttl                  INT DEFAULT NULL,
    prio                 INT DEFAULT NULL,
    change_date          INT DEFAULT NULL,
    disabled             TINYINT(1) DEFAULT 0,
    ordername            VARCHAR(255) BINARY DEFAULT NULL,
    auth                 TINYINT(1) DEFAULT 1,
    PRIMARY KEY (id)
) Engine=InnoDB;

CREATE INDEX nametype_index ON records(name,type);
CREATE INDEX domain_id ON records(domain_id);
CREATE INDEX recordorder ON records (domain_id, ordername);

CREATE TABLE supermasters (
    ip                   VARCHAR(64) NOT NULL,
    nameserver           VARCHAR(255) NOT NULL,
    account              VARCHAR(40) NOT NULL,
    PRIMARY KEY (ip, nameserver)
) Engine=InnoDB;

CREATE TABLE comments (
    id                   INT AUTO_INCREMENT,
    domain_id            INT NOT NULL,
    name                 VARCHAR(255) NOT NULL,
    type                 VARCHAR(10) NOT NULL,
    modified_at          INT NOT NULL,
    account              VARCHAR(40) NOT NULL,
    -- Changed to "TEXT", as VARCHAR(65000) is too big for most MySQL installs
    comment              TEXT NOT NULL,
    PRIMARY KEY (id)

```

```

) Engine=InnoDB;

CREATE INDEX comments_domain_id_idx ON comments (domain_id);
CREATE INDEX comments_name_type_idx ON comments (name, type);
CREATE INDEX comments_order_idx ON comments (domain_id, modified_at);

CREATE TABLE domainmetadata (
  id          INT AUTO_INCREMENT,
  domain_id   INT NOT NULL,
  kind        VARCHAR(32),
  content     TEXT,
  PRIMARY KEY (id)
) Engine=InnoDB;

CREATE INDEX domainmetadata_idx ON domainmetadata (domain_id, kind);

CREATE TABLE cryptokeys (
  id          INT AUTO_INCREMENT,
  domain_id   INT NOT NULL,
  flags       INT NOT NULL,
  active      BOOL,
  content     TEXT,
  PRIMARY KEY(id)
) Engine=InnoDB;

CREATE INDEX domainidindex ON cryptokeys(domain_id);

CREATE TABLE tsigkeys (
  id          INT AUTO_INCREMENT,
  name        VARCHAR(255),
  algorithm   VARCHAR(50),
  secret      VARCHAR(255),
  PRIMARY KEY (id)
) Engine=InnoDB;

CREATE UNIQUE INDEX namealgorithindex ON tsigkeys(name, algorithm);

```

5. Create the PowerDNS schema for the database using `mysql` CLI. For example:

```
root # mysql powerdns < powerdns-schema.sql
```

6. Enable `pdns` `systemd` service.

```
root # systemctl enable pdns
```

```
root # systemctl start pdns
```

If `pdns` is successfully running, you should see the following logs by running `journalctl -u pdns` command.

```
Feb 07 01:44:12 d52-54-77-77-01-01 systemd[1]: Started PowerDNS Authoritative Server.
Feb 07 01:44:12 d52-54-77-77-01-01 pdns_server[21285]: Done launching threads, ready to distribute questions
```

12.1.1.2 Configure Designate To Use PowerDNS Backend

Configure Designate to use PowerDNS backend by appending the PowerDNS servers to `/etc/designate/pools.crowbar.yaml` file on a Designate worker node.



Note

If we are replacing Bind9 backend with PowerDNS backend, make sure to remove the `bind9` entries from `/etc/designate/pools.crowbar.yaml`.

In HA deployment, there should be multiple PowerDNS entries.

Also, make sure the `api_token` matches the `api-key` that was specified in the `/etc/pdns/pdns.conf` file earlier.

Append the PowerDNS entries to the end of `/etc/designate/pools.crowbar.yaml`. For example:

```
---
- name: default-bind
  description: Default BIND9 Pool
  id: 794ccc2c-d751-44fe-b57f-8894c9f5c842
  attributes: {}
  ns_records:
  - hostname: public-d52-54-77-77-01-01.virtual.cloud.suse.de.
    priority: 1
  - hostname: public-d52-54-77-77-01-02.virtual.cloud.suse.de.
    priority: 1
  nameservers:
  - host: 192.168.124.83
    port: 53
  - host: 192.168.124.81
    port: 53
```



```

also_notifies: []
targets:
- type: bind9
  description: BIND9 Server
  masters:
  - host: 192.168.124.83
    port: 5354
  - host: 192.168.124.82
    port: 5354
  - host: 192.168.124.81
    port: 5354
  options:
    host: 192.168.124.83
    port: 53
    rndc_host: 192.168.124.83
    rndc_port: 953
    rndc_key_file: "/etc/designate/rndc.key"
- type: bind9
  description: BIND9 Server
  masters:
  - host: 192.168.124.83
    port: 5354
  - host: 192.168.124.82
    port: 5354
  - host: 192.168.124.81
    port: 5354
  options:
    host: 192.168.124.81
    port: 53
    rndc_host: 192.168.124.81
    rndc_port: 953
    rndc_key_file: "/etc/designate/rndc.key"
- type: pdns4
  description: PowerDNS4 DNS Server
  masters:
  - host: 192.168.124.83
    port: 5354
  - host: 192.168.124.82
    port: 5354
  - host: 192.168.124.81
    port: 5354
  options:
    host: 192.168.124.83
    port: 54
    api_endpoint: http://192.168.124.83:8081
    api_token: Sfw234sDFw90z

```

Update the pools using `designate-manage CLI`.

```
tux > designate-manage pool update --file /etc/designate/pools.crowbar.yaml
```

Once Designate sync up with PowerDNS, you should see the domains in the PowerDNS database which reflects the zones in Designate.



Note

It make take a few minutes for Designate to sync with PowerDNS.

We can verify that the domains are successfully sync up with Designate by inpscting the domains table in the database. For example:

```
root # mysql powerdns
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 21131
Server version: 10.2.29-MariaDB-log SUSE package

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [powerdns]> select * from domains;
+----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
| id | name      | master
last_check | type | notified_serial | account |
+----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
|  1 | foo.bar  | 192.168.124.81:5354 192.168.124.82:5354 192.168.124.83:5354 |
NULL | SLAVE |          NULL |          |
+----+-----+-----+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

12.2 Deploying Pacemaker (Optional, HA Setup Only)

To make the SUSE OpenStack Cloud controller functions and the Compute Nodes highly available, set up one or more clusters by deploying Pacemaker (see [Section 2.6, “High Availability”](#) for details). Since it is possible (and recommended) to deploy more than one cluster, a separate proposal needs to be created for each cluster.

Deploying Pacemaker is optional. In case you do not want to deploy it, skip this section and start the node deployment by deploying the database as described in [Section 12.3, “Deploying the Database”](#).



Note: Number of Cluster Nodes

To set up a cluster, at least two nodes are required. See [Section 2.6.5, “Cluster Requirements and Recommendations”](#) for more information.

To create a proposal, go to *Barclamps > OpenStack* and click *Edit* for the Pacemaker barclamp. A drop-down box where you can enter a name and a description for the proposal opens. Click *Create* to open the configuration screen for the proposal.

●	Pacemaker	Deploy Pacemaker clusters	Edit
+	proposal_1	Created on Fri, 12 Apr 2019 17:16:51 +0000	Create



Important: Proposal Name

The name you enter for the proposal will be used to generate host names for the virtual IP addresses of HAProxy. By default, the names follow this scheme:

cluster-*PROPOSAL_NAME*.*FQDN* (for the internal name)

public-cluster-*PROPOSAL_NAME*.*FQDN* (for the public name)

For example, when *PROPOSAL_NAME* is set to data, this results in the following names:

cluster-data.example.com

public-cluster-data.example.com

For requirements regarding SSL encryption and certificates, see [Section 2.3, “SSL Encryption”](#).

The following options are configurable in the Pacemaker configuration screen:

Transport for Communication

Choose a technology used for cluster communication. You can choose between *Multicast (UDP)*, sending a message to multiple destinations, or *Unicast (UDPU)*, sending a message to a single destination. By default unicast is used.

Policy when cluster does not have quorum

Whenever communication fails between one or more nodes and the rest of the cluster a “cluster partition” occurs. The nodes of a cluster are split in partitions but are still active. They can only communicate with nodes in the same partition and are unaware of the separated nodes. The cluster partition that has the majority of nodes is defined to have “quorum”.

This configuration option defines what to do with the cluster partition(s) that do not have the quorum. See <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#sec-conf-hawk2-cluster-config>, for details.

The recommended setting is to choose *Stop*. However, *Ignore* is enforced for two-node clusters to ensure that the remaining node continues to operate normally in case the other node fails. For clusters using shared resources, choosing *freeze* may be used to ensure that these resources continue to be available.

STONITH: Configuration mode for STONITH

“Misbehaving” nodes in a cluster are shut down to prevent them from causing trouble. This mechanism is called STONITH (“Shoot the other node in the head”). STONITH can be configured in a variety of ways, refer to <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#cha-ha-fencing> for details. The following configuration options exist:

Configured manually

STONITH will not be configured when deploying the barclamp. It needs to be configured manually as described in <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#cha-ha-fencing>. For experts only.

Configured with IPMI data from the IPMI barclamp

Using this option automatically sets up STONITH with data received from the IPMI barclamp. Being able to use this option requires that IPMI is configured for all cluster nodes. This should be done by default. To check or change the IPMI deployment, go to *Barclamps > Crowbar > IPMI > Edit*. Also make sure the *Enable BMC* option is set to *true* on this barclamp.



Important: STONITH Devices Must Support IPMI

To configure STONITH with the IPMI data, *all* STONITH devices must support IPMI. Problems with this setup may occur with IPMI implementations that are not strictly standards compliant. In this case it is recommended to set up STONITH with STONITH block devices (SBD).

Configured with STONITH Block Devices (SBD)

This option requires manually setting up shared storage and a watchdog on the cluster nodes before applying the proposal. To do so, proceed as follows:

1. Prepare the shared storage. The path to the shared storage device must be persistent and consistent across all nodes in the cluster. The SBD device must not use host-based RAID or cLVM2.
2. Install the package `sbd` on all cluster nodes.
3. Initialize the SBD device with by running the following command. Make sure to replace `/dev/SBD` with the path to the shared storage device.

```
sbd -d /dev/SBD create
```

Refer to <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#sec-ha-storage-protect-test> for details.

In *Kernel module for watchdog*, specify the respective kernel module to be used. Find the most commonly used watchdog drivers in the following table:

Hardware	Driver
HP	<code>hpwdt</code>
Dell, Fujitsu, Lenovo (Intel TCO)	<code>iTCO_wdt</code>
Generic	<code>softdog</code>

If your hardware is not listed above, either ask your hardware vendor for the right name or check the following directory for a list of choices: `/lib/modules/KERNEL_VERSION/kernel/drivers/watchdog`.

Alternatively, list the drivers that have been installed with your kernel version:

```
root # rpm -ql kernel-VERSION | grep watchdog
```

If the nodes need different watchdog modules, leave the text box empty.

After the shared storage has been set up, specify the path using the “by-id” notation (`/dev/disk/by-id/DEVICE`). It is possible to specify multiple paths as a comma-separated list.

Deploying the barclamp will automatically complete the SBD setup on the cluster nodes by starting the SBD daemon and configuring the fencing resource.

Configured with one shared resource for the whole cluster

All nodes will use the identical configuration. Specify the *Fencing Agent* to use and enter *Parameters* for the agent.

To get a list of STONITH devices which are supported by the High Availability Extension, run the following command on an already installed cluster nodes: `stonith -L`. The list of parameters depends on the respective agent. To view a list of parameters use the following command:

```
stonith -t agent -n
```

Configured with one resource per node

All nodes in the cluster use the same *Fencing Agent*, but can be configured with different parameters. This setup is, for example, required when nodes are in different chassis and therefore need different IPMI parameters.

To get a list of STONITH devices which are supported by the High Availability Extension, run the following command on an already installed cluster nodes: `stonith -L`. The list of parameters depends on the respective agent. To view a list of parameters use the following command:

```
stonith -t agent -n
```

Configured for nodes running in libvirt

Use this setting for completely virtualized test installations. This option is not supported.

STONITH: Do not start corosync on boot after fencing

With STONITH, Pacemaker clusters with two nodes may sometimes hit an issue known as STONITH deathmatch where each node kills the other one, resulting in both nodes rebooting all the time. Another similar issue in Pacemaker clusters is the fencing loop, where a reboot caused by STONITH will not be enough to fix a node and it will be fenced again and again.

This setting can be used to limit these issues. When set to *true*, a node that has not been properly shut down or rebooted will not start the services for Pacemaker on boot. Instead, the node will wait for action from the SUSE OpenStack Cloud operator. When set to *false*, the services for Pacemaker will always be started on boot. The *Automatic* value is used to have the most appropriate value automatically picked: it will be *true* for two-node clusters (to avoid STONITH deathmatches), and *false* otherwise.

When a node boots but not starts corosync because of this setting, then the node's status is in the *Node Dashboard* is set to "Problem" (red dot).

Mail Notifications: Enable Mail Notifications

Get notified of cluster node failures via e-mail. If set to *true*, you need to specify which *SMTP Server* to use, a prefix for the mails' subject and sender and recipient addresses. Note that the SMTP server must be accessible by the cluster nodes.

HAProxy: Public name for public virtual IP

The public name is the host name that will be used instead of the generated public name (see *Important: Proposal Name*) for the public virtual IP address of HAProxy. (This is the case when registering public endpoints, for example). Any name specified here needs to be resolved by a name server placed outside of the SUSE OpenStack Cloud network.

● Edit Proposal

Pacemaker: Proposal 1 !Attributes Raw

Corosync Rings

Transport for Communication

Unicast (UDPU) ▾

Ring 1

Network

admin

Ring 2 Add Second Ring

Network

A network defined in the network barclamp (admin, storage, etc.)

Pacemaker Parameters

Policy when cluster does not have quorum

ignore ▾

Refer to the [High Availability Guide](#) for a description of each value.

FIGURE 12.1: THE PACEMAKER BARCLAMP

The Pacemaker component consists of the following roles. Deploying the *hawk-server* role is optional:

pacemaker-cluster-member

Deploy this role on all nodes that should become member of the cluster.

hawk-server

Deploying this role is optional. If deployed, sets up the Hawk Web interface which lets you monitor the status of the cluster. The Web interface can be accessed via <https://IP-ADDRESS:7630>. The default hawk credentials are username `hacluster`, password `crowbar`. The password is visible and editable in the *Custom* view of the Pacemaker barclamp, and also in the "`corosync`": section of the *Raw* view.

Note that the GUI on SUSE OpenStack Cloud can only be used to monitor the cluster status and not to change its configuration.

hawk-server may be deployed on at least one cluster node. It is recommended to deploy it on all cluster nodes.

pacemaker-remote

Deploy this role on all nodes that should become members of the Compute Nodes cluster. They will run as Pacemaker remote nodes that are controlled by the cluster, but do not affect quorum. Instead of the complete cluster stack, only the `pacemaker-remote` component will be installed on this nodes.

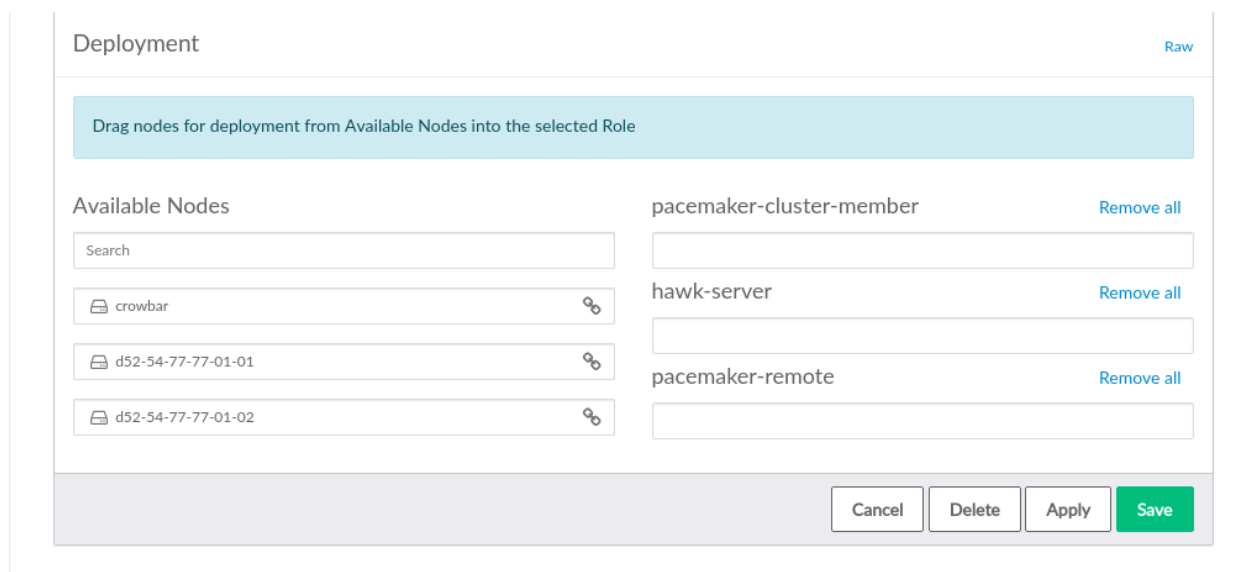


FIGURE 12.2: THE PACEMAKER BARCLAMP: NODE DEPLOYMENT EXAMPLE

After a cluster has been successfully deployed, it is listed under *Available Clusters* in the *Deployment* section and can be used for role deployment like a regular node.



Warning: Deploying Roles on Single Cluster Nodes

When using clusters, roles from other barclamps must never be deployed to single nodes that are already part of a cluster. The only exceptions from this rule are the following roles:

- `cinder-volume`
- `swift-proxy` + `swift-dispersion`

- swift-ring-compute
- swift-storage

Important: Service Management on the Cluster

After a role has been deployed on a cluster, its services are managed by the HA software. You must *never* manually start or stop an HA-managed service, nor configure it to start on boot. Services may only be started or stopped by using the cluster management tools Hawk or the `crm` shell. See <https://documentation.suse.com/sle-ha/15-SP1/single-html/SLE-HA-guide/#sec-ha-config-basics-resources> for more information.

Note: Testing the Cluster Setup

To check whether all cluster resources are running, either use the Hawk Web interface or run the command `crm_mon -lr`. If it is not the case, clean up the respective resource with `crm resource cleanup RESOURCE`, so it gets respawned.

Also make sure that STONITH correctly works before continuing with the SUSE OpenStack Cloud setup. This is especially important when having chosen a STONITH configuration requiring manual setup. To test if STONITH works, log in to a node on the cluster and run the following command:

```
pkill -9 corosync
```

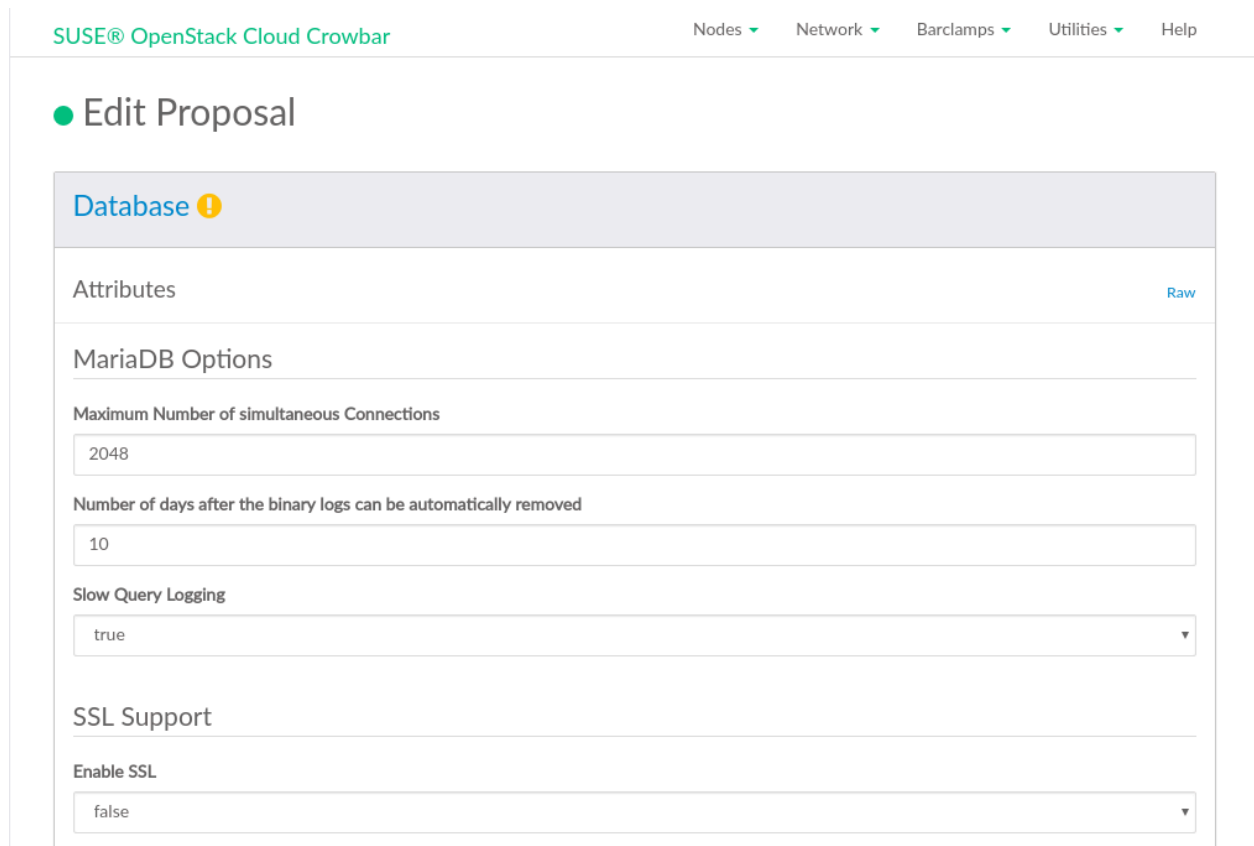
In case STONITH is correctly configured, the node will reboot.

Before testing on a production cluster, plan a maintenance window in case issues should arise.

12.3 Deploying the Database

The very first service that needs to be deployed is the *Database*. The database component is using MariaDB and is used by all other components. It must be installed on a Control Node. The Database can be made highly available by deploying it on a cluster.

The only attribute you may change is the maximum number of database connections (*Global Connection Limit*). The default value should usually work—only change it for large deployments in case the log files show database connection failures.



SUSE® OpenStack Cloud Crowbar

Nodes ▾ Network ▾ Barclamps ▾ Utilities ▾ Help

● Edit Proposal

Database ⚠

Attributes Raw

MariaDB Options

Maximum Number of simultaneous Connections

Number of days after the binary logs can be automatically removed

Slow Query Logging

SSL Support

Enable SSL

FIGURE 12.3: THE DATABASE BARCLAMP

12.3.1 Deploying MariaDB

Deploying the database requires the use of MariaDB



Note: MariaDB and HA

MariaDB back end features full HA support based on the Galera clustering technology. The HA setup requires an odd number of nodes. The recommended number of nodes is 3.

12.3.1.1 SSL Configuration

SSL can be enabled with either a stand-alone or cluster deployment. The replication traffic between database nodes is not encrypted, whilst traffic between the database server(s) and clients are, so a separate network for the database servers is recommended.

Certificates can be provided, or the barclamp can generate self-signed certificates. The certificate filenames are configurable in the barclamp, and the directories `/etc/mysql/ssl/certs` and `/etc/mysql/ssl/private` to use the defaults will need to be created before the barclamp is applied. The CA certificate and the certificate for MariaDB to use both go into `/etc/mysql/ssl/certs`. The appropriate private key for the certificate is placed into the `/etc/mysql/ssl/private` directory. As long as the files are readable when the barclamp is deployed, permissions can be tightened after a successful deployment once the appropriate UNIX groups exist.

The Common Name (CN) for the SSL certificate must be `fully qualified server name` for single host deployments, and `cluster-cluster name.full domain name` for cluster deployments.

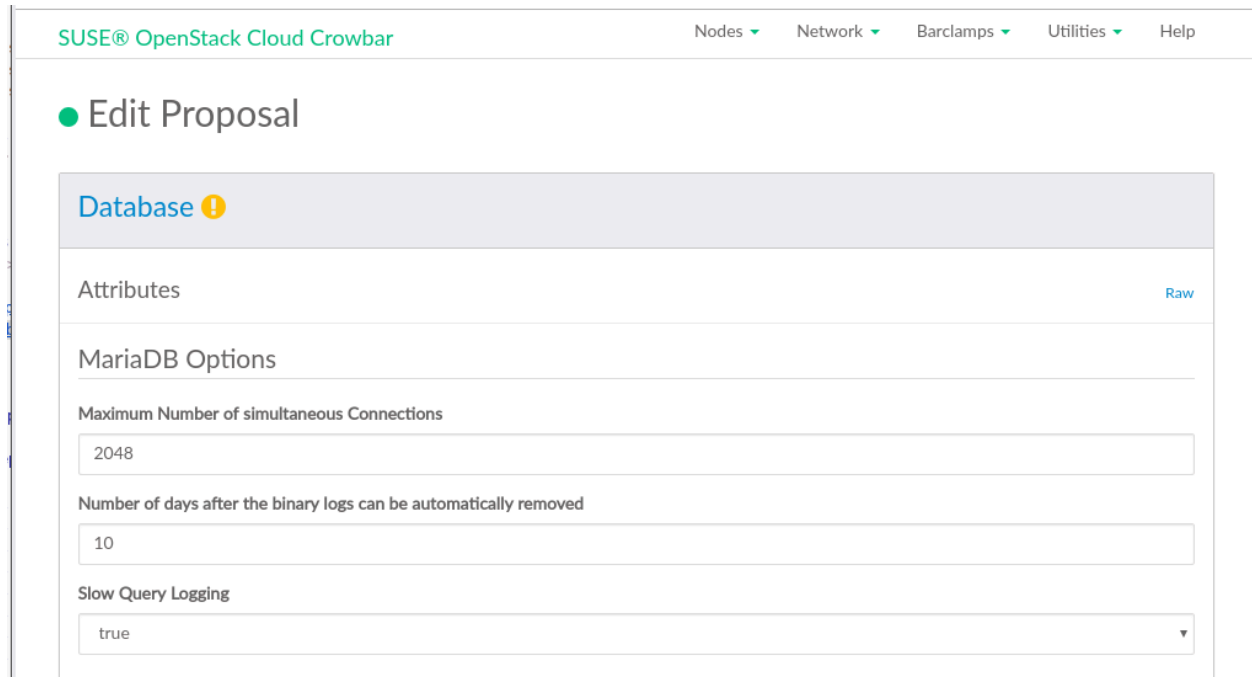


Note: Certificate validation errors

If certificate validation errors are causing issues with deploying other barclamps (for example, when creating databases or users) you can check the configuration with `mysql --ssl-verify-server-cert` which will perform the same verification that Crowbar does when connecting to the database server.

If certificates are supplied, the CA certificate and its full trust chain must be in the `ca.pem` file. The certificate must be trusted by the machine (or all cluster members in a cluster deployment), and it must be available on all client machines — IE, if the OpenStack services are deployed on separate machines or cluster members they will all require the CA certificate to be in `/etc/mysql/ssl/certs` as well as trusted by the machine.

12.3.1.2 MariaDB Configuration Options



The screenshot shows the 'Edit Proposal' page for the 'Database' barclamp. The page title is 'SUSE® OpenStack Cloud Crowbar' with navigation links for 'Nodes', 'Network', 'Barclamps', 'Utilities', and 'Help'. The main heading is '● Edit Proposal'. Below this is a section titled 'Database' with a warning icon. Underneath is a table with 'Attributes' and a 'Raw' link. The 'MariaDB Options' section contains three configuration fields: 'Maximum Number of simultaneous Connections' with a value of 2048, 'Number of days after the binary logs can be automatically removed' with a value of 10, and 'Slow Query Logging' with a value of true.

FIGURE 12.4: MARIADB CONFIGURATION

The following configuration settings are available via the *Database* barclamp graphical interface:

Datadir

Path to a directory for storing database data.

Maximum Number of Simultaneous Connections

The maximum number of simultaneous client connections.

Number of days after the binary logs can be automatically removed

A period after which the binary logs are removed.

Slow Query Logging

When enabled, all queries that take longer than usual to execute are logged to a separate log file (by default, it's `/var/log/mysql/mysql_slow.log`). This can be useful for debugging.



Warning: MariaDB Deployment Restriction

When MariaDB is used as the database back end, the *monasca-server* role cannot be deployed to the node with the *database-server* role. These two roles cannot coexist due to the fact that monasca uses its own MariaDB instance.

12.4 Deploying RabbitMQ

The RabbitMQ messaging system enables services to communicate with the other nodes via Advanced Message Queue Protocol (AMQP). Deploying it is mandatory. RabbitMQ needs to be installed on a Control Node. RabbitMQ can be made highly available by deploying it on a cluster. We recommend not changing the default values of the proposal's attributes.

Virtual Host

Name of the default virtual host to be created and used by the RabbitMQ server (default_vhost configuration option in rabbitmq.config).

Port

Port the RabbitMQ server listens on (tcp_listeners configuration option in rabbitmq.config).

User

RabbitMQ default user (default_user configuration option in rabbitmq.config).

● Edit Proposal

RabbitMQ ✓

Attributes Raw

Virtual host

Port

User

Configure Clients to send notifications

Extra users

Username	Permissions (3 comma separated items for configure, write, read; e.g. ".*, *, **")	Tags (comma separated)
No records		

SSL Support

Enable SSL

Deployment Raw

Drag nodes for deployment from Available Nodes into the selected Role

Available Nodes

- crowbar ⊞
- d52-54-77-77-01-01 ⊞
- d52-54-77-77-01-02 ⊞

rabbitmq-server Remove all

 ⊞

FIGURE 12.5: THE RABBITMQ BARCLAMP

12.4.1 HA Setup for RabbitMQ

To make RabbitMQ highly available, deploy it on a cluster instead of a single Control Node. This also requires shared storage for the cluster that hosts the RabbitMQ data. We recommend using a dedicated cluster to deploy RabbitMQ together with the database, since both components require shared storage.

Deploying RabbitMQ on a cluster makes an additional *High Availability* section available in the *Attributes* section of the proposal. Configure the *Storage Mode* in this section.

12.4.2 SSL Configuration for RabbitMQ

The RabbitMQ barclamp supports securing traffic via SSL. This is similar to the SSL support in other barclamps, but with these differences:

- RabbitMQ can listen on two ports at the same time, typically port 5672 for unsecured and port 5671 for secured traffic.
- The ceilometer pipeline for OpenStack swift cannot be passed SSL-related parameters. When SSL is enabled for RabbitMQ the ceilometer pipeline in swift is turned off, rather than sending it over an unsecured channel.

The following steps are the fastest way to set up and test a new SSL certificate authority (CA).

1. In the RabbitMQ barclamp set *Enable SSL* to *true*, and *Generate (self-signed) certificates (implies insecure)* to `true`, then apply the barclamp. The barclamp will create a new CA, enter the correct settings in `/etc/rabbitmq/rabbitmq.config`, and start RabbitMQ.
2. Test your new CA with OpenSSL, substituting the hostname of your control node:

```
openssl s_client -connect d52-54-00-59-e5-fd:5671
[...]
Verify return code: 18 (self signed certificate)
```

This outputs a lot of information, including a copy of the server's public certificate, protocols, ciphers, and the chain of trust.

3. The last step is to configure client services to use SSL to access the RabbitMQ service. (See <https://www.rabbitmq.com/ssl.html> for a complete reference).

It is preferable to set up your own CA. The best practice is to use a commercial certificate authority. You may also deploy your own self-signed certificates, provided that your cloud is not publicly-accessible, and only for your internal use. Follow these steps to enable your own CA in RabbitMQ and deploy it to SUSE OpenStack Cloud:

- Configure the RabbitMQ barclamp to use the control node's certificate authority (CA), if it already has one, or create a CA specifically for RabbitMQ and configure the barclamp to use that. (See [Section 2.3, “SSL Encryption”](#), and the RabbitMQ manual has a detailed howto on creating your CA at <http://www.rabbitmq.com/ssl.html>, with customizations for .NET and Java clients.)

The screenshot shows the 'SSL Support' configuration section of the RabbitMQ barclamp. It contains several fields and dropdown menus:

- Enable SSL:** A dropdown menu set to 'true'.
- SSL Port:** A text input field containing '5671'.
- Generate (self-signed) certificates (implies insecure):** A dropdown menu set to 'false'.
- SSL Certificate File:** A text input field containing '/etc/rabbitmq/ssl/certs/signing_cert.pem'.
- SSL (Private) Key File:** A text input field containing '/etc/rabbitmq/ssl/private/signing_key.pem'.
- Require Client Certificate:** A dropdown menu set to 'false'.
- SSL CA Certificates File:** A text input field containing '/etc/rabbitmq/ssl/certs/ca.pem'.
- SSL Certificate is insecure (for instance, self-signed):** A dropdown menu set to 'false'.
- SSL client CA file (used to validate rabbitmq server certificate):** A text input field containing '/etc/ssl/certs/rabbitca.pem'.

FIGURE 12.6: SSL SETTINGS FOR RABBITMQ BARCLAMP

The configuration options in the RabbitMQ barclamp allow tailoring the barclamp to your SSL setup.

Enable SSL

Set this to *True* to expose all of your configuration options.

SSL Port

RabbitMQ's SSL listening port. The default is 5671.

Generate (self-signed) certificates (implies insecure)

When this is set to `true`, self-signed certificates are automatically generated and copied to the correct locations on the control node, and all other barclamp options are set automatically. This is the fastest way to apply and test the barclamp. Do not use this on production systems. When this is set to `false` the remaining options are exposed.

SSL Certificate File

The location of your public root CA certificate.

SSL (Private) Key File

The location of your private server key.

Require Client Certificate

This goes with *SSL CA Certificates File*. Set to `true` to require clients to present SSL certificates to RabbitMQ.

SSL CA Certificates File

Trust client certificates presented by the clients that are signed by other CAs. You'll need to store copies of the CA certificates; see "Trust the Client's Root CA" at <http://www.rabbitmq.com/ssl.html>.

SSL Certificate is insecure (for instance, self-signed)

When this is set to `false`, clients validate the RabbitMQ server certificate with the *SSL client CA file*.

SSL client CA file (used to validate rabbitmq server certificate)

Tells clients of RabbitMQ where to find the CA bundle that validates the certificate presented by the RabbitMQ server, when *SSL Certificate is insecure (for instance, self-signed)* is set to `false`.

12.4.3 Configuring Clients to Send Notifications

RabbitMQ has an option called `Configure clients to send notifications`. It defaults to `false`, which means no events will be sent. It is required to be set to `true` for ceilometer, monasca, and any other services consuming notifications. When it is set to `true`, OpenStack services are configured to submit lifecycle audit events to the `notification` RabbitMQ queue.

This option should only be enabled if an active consumer is configured, otherwise events will accumulate on the RabbitMQ server, clogging up CPU, memory, and disk storage.

Any accumulation can be cleared by running:

```
$ rabbitmqctl -p /openstack purge_queue notifications.info
$ rabbitmqctl -p /openstack purge_queue notifications.error
```

12.5 Deploying keystone

keystone is another core component that is used by all other OpenStack components. It provides authentication and authorization services. keystone needs to be installed on a Control Node. keystone can be made highly available by deploying it on a cluster. You can configure the following parameters of this barclamp:

Algorithm for Token Generation

Set the algorithm used by keystone to generate the tokens. You can choose between Fernet (the default) or UUID. Note that for performance and security reasons it is strongly recommended to use Fernet.

Region Name

Allows customizing the region name that crowbar is going to manage.

Default Credentials: Default Tenant

Tenant for the users. Do not change the default value of openstack.

Default Credentials: Administrator User Name/Password

User name and password for the administrator.

Default Credentials: Create Regular User

Specify whether a regular user should be created automatically. Not recommended in most scenarios, especially in an LDAP environment.

Default Credentials: Regular User Username/Password

User name and password for the regular user. Both the regular user and the administrator accounts can be used to log in to the SUSE OpenStack Cloud Dashboard. However, only the administrator can manage keystone users and access.

● Edit Proposal

Keystone !

Attributes Raw

Region name

Default Credentials

Default Project

Administrator Username

Administrator Password

 👁

Create Regular User

Regular User Username

Regular User Password

 👁

SSL Support

Protocol

FIGURE 12.7: THE KEYSTONE BARCLAMP

SSL Support: Protocol

When you use the default value *HTTP*, public communication will not be encrypted. Choose *HTTPS* to use SSL for encryption. See [Section 2.3, “SSL Encryption”](#) for background information and [Section 11.4.6, “Enabling SSL”](#) for installation instructions. The following additional configuration options will become available when choosing *HTTPS*:

Generate (self-signed) certificates

When set to `true`, self-signed certificates are automatically generated and copied to the correct locations. This setting is for testing purposes only and should never be used in production environments!

SSL Certificate File / SSL (Private) Key File

Location of the certificate key pair files.

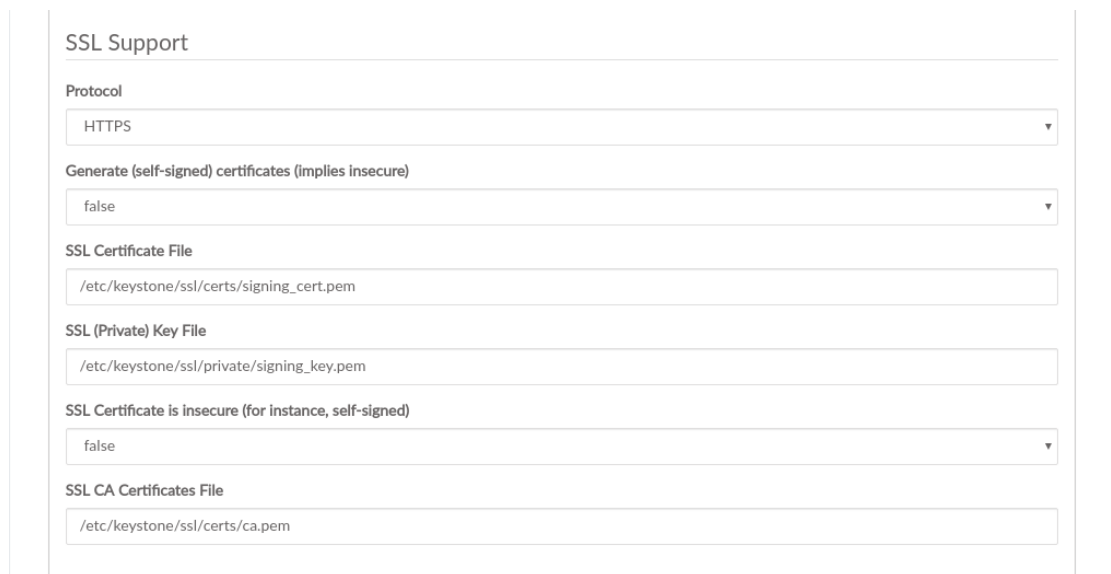
SSL Certificate is insecure

Set this option to `true` when using self-signed certificates to disable certificate checks. This setting is for testing purposes only and should never be used in production environments!

SSL CA Certificates File

Specify the absolute path to the CA certificate. This field is mandatory, and leaving it blank will cause the barclamp to fail. To fix this issue, you have to provide the absolute path to the CA certificate, restart the `apache2` service, and re-deploy the barclamp.

When the certificate is not already trusted by the pre-installed list of trusted root certificate authorities, you need to provide a certificate bundle that includes the root and all intermediate CAs.



The screenshot shows a configuration dialog titled "SSL Support". It contains several fields and dropdown menus:

- Protocol:** A dropdown menu with "HTTPS" selected.
- Generate (self-signed) certificates (implies insecure):** A dropdown menu with "false" selected.
- SSL Certificate File:** A text input field containing the path `/etc/keystone/ssl/certs/signing_cert.pem`.
- SSL (Private) Key File:** A text input field containing the path `/etc/keystone/ssl/private/signing_key.pem`.
- SSL Certificate is insecure (for instance, self-signed):** A dropdown menu with "false" selected.
- SSL CA Certificates File:** A text input field containing the path `/etc/keystone/ssl/certs/ca.pem`.

FIGURE 12.8: THE SSL DIALOG

12.5.1 Authenticating with LDAP

keystone has the ability to separate identity back-ends by domains. SUSE OpenStack Cloud 9 uses this method for authenticating users.

The keystone barclamp sets up a MariaDB database by default. Configuring an LDAP back-end is done in the *Raw* view.

1. Set `"domain_specific_drivers": true`,
2. Then in the `"domain_specific_config"`: section configure a map with domain names as keys, and configuration as values. In the default proposal the domain name key is `"ldap_users"`, and the keys are the two required sections for an LDAP-based identity driver configuration, the `[identity]` section which sets the driver, and the `[ldap]` section which sets the LDAP connection options. You may configure multiple domains, each with its own configuration.

You may make this available to horizon by setting `multi_domain_support` to `true` in the horizon barclamp.

Users in the LDAP-backed domain have to know the name of the domain in order to authenticate, and must use the keystone v3 API endpoint. (See the OpenStack manuals, [Domain-specific Configuration \(https://docs.openstack.org/keystone/rocky/admin/identity-domain-specific-config.html\)](https://docs.openstack.org/keystone/rocky/admin/identity-domain-specific-config.html) and [Integrate Identity with LDAP \(https://docs.openstack.org/keystone/rocky/admin/identity-integrate-with-ldap.html\)](https://docs.openstack.org/keystone/rocky/admin/identity-integrate-with-ldap.html), for additional details.)

12.5.2 HA Setup for keystone

Making keystone highly available requires no special configuration—it is sufficient to deploy it on a cluster.

12.5.3 OpenID Connect Setup for keystone

keystone supports WebSSO by federating with an external identity provider (IdP) using `auth_openidc` (https://github.com/zmartzone/mod_auth_openidc) module.

There are two steps to enable this feature:

1. Configure the `"federation"` and `"openidc"` attributes for the Keystone Barclamp in Crowbar.
2. Create the Identity Provider, Protocol, and Mapping resource in keystone using *OpenStack Command Line Tool (CLI)*.

12.5.3.1 keystone Barclamp Configuration

Configuring OpenID Connect is done in the *Raw* view, under the *federation* section. The global attributes, namely *trusted_dashboards* and *websso_keystone_url* are not specific to OpenID Connect. Rather, they are designed to help facilitate WebSSO browser redirects with external IdPs in a complex cloud deployment environment.

If the cloud deployment does not have any external proxies or load balancers, where the public keystone and horizon (Dashboard service) endpoints are directly managed by Crowbar, *trusted_dashboards* and *websso_keystone_url* does not need to be provided. However, in a complex cloud deployment where the public Keystone and Horizon endpoints are handled by external load balancers or proxies, and they are not directly managed by Crowbar, *trusted_dashboards* and *websso_keystone_url* must be provided and they must correctly reflect the external public endpoints. To configure OpenID Connect, edit the attributes in the *openidc* subsection.

1. Set *"enabled":true*
2. Provide the name for the *identity_provider*. This must be the same as the identity provider to be created in Keystone using the *OpenStack Command Line Tool (CLI)*. For example, if the identity provider is *foo*, create the identity provider with the name via Openstack CLI (i.e. *openstack identity provider create foo*).
3. *response_type* corresponds to *auth_openidc OIDCResponseType* (https://github.com/zmartzone/mod_auth_openidc/blob/master/auth_openidc.conf). In most cases, it should be *id_token*.
4. *scope* corresponds to *auth_openidc OIDCScope* (https://github.com/zmartzone/mod_auth_openidc/blob/master/auth_openidc.conf).
5. *metadata_url* corresponds to *auth_openidc OIDCProviderMetadataURL* (https://github.com/zmartzone/mod_auth_openidc/blob/master/auth_openidc.conf).
6. *client_id* corresponds to *auth_openidc OIDCClientID* (https://github.com/zmartzone/mod_auth_openidc/blob/master/auth_openidc.conf).
7. *client_secret* corresponds to *auth_openidc OIDCClientSecret* (https://github.com/zmartzone/mod_auth_openidc/blob/master/auth_openidc.conf).
8. *redirect_uri* corresponds to *auth_openidc OIDCRedirectURI* (https://github.com/zmartzone/mod_auth_openidc/blob/master/auth_openidc.conf). In a cloud deployment where all the external endpoints are directly managed by Crowbar, this attribute can be left

blank as it will be auto-populated by Crowbar. However, in a complex cloud deployment where the public Keystone endpoint is handled by an external load balancer or proxy, this attribute must reflect the external Keystone auth endpoint for the OpenID Connect IdP. For example, "https://keystone-public-endpoint.foo.com/v3/OS-FEDERATION/identity_providers/foo/protocols/openid/auth"



Warning

Some OpenID Connect IdPs such as Google require the hostname in the *redirect_uri* to be a public FQDN. In that case, the hostname in Keystone public endpoint must also be a public FQDN and must match the one specified in the *redirect_uri*.

12.5.3.2 Create Identity Provider, Protocol, and Mapping

To fully enable OpenID Connect, the Identity Provider, Protocol, and Mapping for the given IdP must be created in Keystone. This is done by using the *OpenStack Command Line Tool*, on a controller node, and using the Keystone **admin** credential.

1. Login to a controller node as root user.
2. Use the Keystone **admin** credential.

```
source ~/.openrc
```

3. Create the **Identity Provider**. For example:

```
openstack identity provider create foo
```



Warning

The name of the Identity Provider must be exactly the same as the *identity_provider* attribute given when configuring Keystone in the previous section.

4. Next, create the Mapping for the Identity Provider. Prior to creating the Mapping, one must fully grasp the intricacies of [Mapping Combinations](https://docs.openstack.org/keystone/latest/admin/federation/mapping_combinations.html) (https://docs.openstack.org/keystone/latest/admin/federation/mapping_combinations.html) [↗](#) as it may have profound security implications if done incorrectly. Here's an example of a mapping file.

```
[
```



```

{
  "local": [
    {
      "user": {
        "name": "{0}",
        "email": "{1}",
        "type": "ephemeral"
      },
      "group": {
        "domain": {
          "name": "Default"
        },
        "name": "openidc_demo"
      }
    }
  ],
  "remote": [
    {
      "type": "REMOTE_USER"
    },
    {
      "type": "HTTP_OIDC_EMAIL"
    }
  ]
}
]

```

Once the mapping file is created, now create the mapping resource in Keystone. For example:

```
openstack mapping create --rule oidc_mapping.json oidc_mapping
```

5. Lastly, create the Protocol for the Identity Provider and its mapping. For OpenID Connect, the protocol name must be **openid**. For example:

```
openstack federation protocol create --identity-provider google --mapping
oidc_mapping openid
```

12.6 Deploying monasca (Optional)

monasca is an open-source monitoring-as-a-service solution that integrates with OpenStack. monasca is designed for scalability, high performance, and fault tolerance.

Accessing the *Raw* interface is not required for day-to-day operation. But as not all monasca settings are exposed in the barclamp graphical interface (for example, various performance tuneables), it is recommended to configure monasca in the *Raw* mode. Below are the options that can be configured via the *Raw* interface of the monasca barclamp.

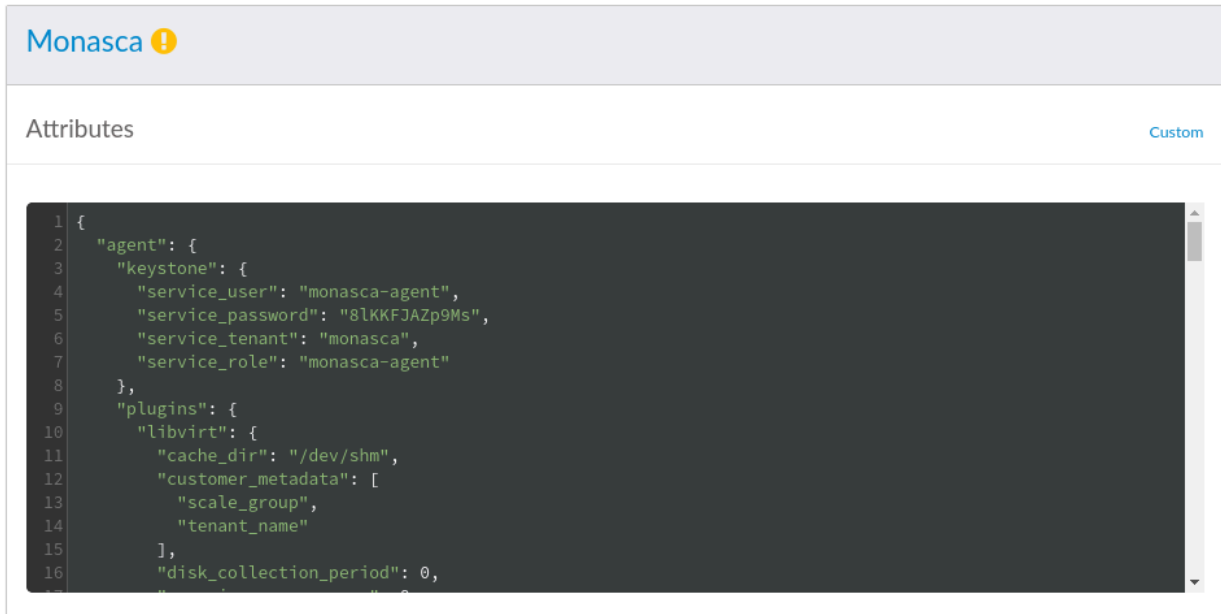


FIGURE 12.9: THE MONASCA BARCLAMP RAW MODE

agent: settings for openstack-monasca-agent

keystone

Contains keystone credentials that the agents use to send metrics. Do not change these options, as they are configured by Crowbar.

insecure

Specifies whether SSL certificates are verified when communicating with keystone. If set to `false`, the `ca_file` option must be specified.

ca_file

Specifies the location of a CA certificate that is used for verifying keystone's SSL certificate.

log_dir

Path for storing log files. The specified path must exist. Do not change the default `/var/log/monasca-agent` path.

log_level

Agent's log level. Limits log messages to the specified level and above. The following levels are available: Error, Warning, Info (default), and Debug.

check_frequency

Interval in seconds between running agents' checks.

num_collector_threads

Number of simultaneous collector threads to run. This refers to the maximum number of different collector plug-ins (for example, `http_check`) that are allowed to run simultaneously. The default value `1` means that plug-ins are run sequentially.

pool_full_max_retries

If a problem with the results from multiple plug-ins results blocks the entire thread pool (as specified by the `num_collector_threads` parameter), the collector exits, so it can be restarted by the `supervisord`. The parameter `pool_full_max_retries` specifies when this event occurs. The collector exits when the defined number of consecutive collection cycles have ended with the thread pool completely full.

plugin_collect_time_warn

Upper limit in seconds for any collection plug-in's run time. A warning is logged if a plug-in runs longer than the specified limit.

max_measurement_buffer_size

Maximum number of measurements to buffer locally if the monasca API is unreachable. Measurements will be dropped in batches, if the API is still unreachable after the specified number of messages are buffered. The default `-1` value indicates unlimited buffering. Note that a large buffer increases the agent's memory usage.

backlog_send_rate

Maximum number of measurements to send when the local measurement buffer is flushed.

amplifier

Number of extra dimensions to add to metrics sent to the monasca API. This option is intended for load testing purposes only. Do not enable the option in production! The default `0` value disables the addition of dimensions.

log_agent: settings for openstack-monasca-log-agent

max_data_size_kb

Maximum payload size in kilobytes for a request sent to the monasca log API.

num_of_logs

Maximum number of log entries the log agent sends to the monasca log API in a single request. Reducing the number increases performance.

elapsed_time_sec

Time interval in seconds between sending logs to the monasca log API.

delay

Interval in seconds for checking whether elapsed_time_sec has been reached.

keystone

keystone credentials the log agents use to send logs to the monasca log API. Do not change this option manually, as it is configured by Crowbar.

api: Settings for openstack-monasca-api

bind_host

Interfaces monasca-api listens on. Do not change this option, as it is configured by Crowbar.

processes

Number of processes to spawn.

threads

Number of WSGI worker threads to spawn.

log_level

Log level for openstack-monasca-api. Limits log messages to the specified level and above. The following levels are available: Critical, Error, Warning, Info (default), Debug, and Trace.

elasticsearch: server-side settings for elasticsearch

repo_dir

List of directories for storing Elasticsearch snapshots. Must be created manually and be writeable by the `elasticsearch` user. Must contain at least one entry in order for the snapshot functionality to work.

heap_size

Sets the heap size. We recommend setting heap size at 50% of the available memory, but not more than 31 GB. The default of 4 GB is likely too small and should be increased if possible.

limit_memlock

The maximum size that may be locked into memory in bytes

limit_nofile

The maximum number of open file descriptors

limit_nproc

The maximum number of processes

vm_max_map_count

The maximum number of memory map areas a process may have.

For instructions on creating an Elasticsearch snapshot, see *Book "Operations Guide Crowbar", Chapter 4 "SUSE OpenStack Cloud Monitoring", Section 4.7 "Operation and Maintenance", Section 4.7.4 "Backup and Recovery"*.

elasticsearch_curator: settings for elastisearch-curator

`elasticsearch-curator` removes old and large elasticsearch indices. The settings below determine its behavior.

delete_after_days

Time threshold for deleting indices. Indices older the specified number of days are deleted. This parameter is unset by default, so indices are kept indefinitely.

delete_after_size

Maximum size in megabytes of indices. Indices larger than the specified size are deleted. This parameter is unset by default, so indices are kept irrespective of their size.

delete_exclude_index

List of indices to exclude from `elasticsearch-curator` runs. By default, only the `.kibana` files are excluded.

kafka: tunables for Kafka

log_retention_hours

Number of hours for retaining log segments in Kafka's on-disk log. Messages older than the specified value are dropped.

log_retention_bytes

Maximum size for Kafka's on-disk log in bytes. If the log grows beyond this size, the oldest log segments are dropped.

topics

list of topics

- metrics
- events
- alarm-state-transitions
- alarm-notifications
- retry-notifications
- 60-seconds-notifications
- log
- transformed-log

The following are options of every topic:

replicas

Controls how many servers replicate each message that is written

partitions

Controls how many logs the topic is sharded into

config_options

Map of configuration options is described in the [Apache Kafka documentation \(https://kafka.apache.org/documentation/#topicconfigs\)](https://kafka.apache.org/documentation/#topicconfigs) ↗

These parameters only affect first time installations. Parameters may be changed after installation with scripts available from [Apache Kafka \(https://kafka.apache.org/documentation/#basic_ops\)](https://kafka.apache.org/documentation/#basic_ops).

Kafka does not support reducing the number of partitions for a topic.

notification:

email_enabled

Enable or disable email alarm notifications.

email_smtp_host

SMTP smarthost for sending alarm notifications.

email_smtp_port

Port for the SMTP smarthost.

email_smtp_user

User name for authenticating against the smarthost.

email_smtp_password

Password for authenticating against the smarthost.

email_smtp_from_address

Sender address for alarm notifications.

master: configuration for monasca-installer on the Crowbar node

influxdb_retention_policy

Number of days to keep metrics records in influxdb.

For an overview of all supported values, see https://docs.influxdata.com/influxdb/v1.1/query_language/database_management/#create-retention-policies-with-create-retention-policy.

monasca: settings for libvirt and Ceph monitoring

monitor_libvirt

The global switch for toggling libvirt monitoring. If set to `true`, libvirt metrics will be gathered on all libvirt based Compute Nodes. This setting is available in the Crowbar UI.

monitor_ceph

The global switch for toggling Ceph monitoring. If set to `true`, Ceph metrics will be gathered on all Ceph-based Compute Nodes. This setting is available in Crowbar UI. If the Ceph cluster has been set up independently, Crowbar ignores this setting.

cache_dir

The directory where monasca-agent will locally cache various metadata about locally running VMs on each Compute Node.

customer_metadata

Specifies the list of instance metadata keys to be included as dimensions with customer metrics. This is useful for providing more information about an instance.

disk_collection_period

Specifies a minimum interval in seconds for collecting disk metrics. Increase this value to reduce I/O load. If the value is lower than the global agent collection period (`check_frequency`), it will be ignored in favor of the global collection period.

max_ping_concurrency

Specifies the number of ping command processes to run concurrently when determining whether the VM is reachable. This should be set to a value that allows the plugin to finish within the agent's collection period, even if there is a networking issue. For example, if the expected number of VMs per Compute Node is 40 and each VM has one IP address, then the plugin will take at least 40 seconds to do the ping checks in the worst-case scenario where all pings fail (assuming the default timeout of 1 second). Increasing `max_ping_concurrency` allows the plugin to finish faster.

metadata

Specifies the list of nova side instance metadata keys to be included as dimensions with the cross-tenant metrics for the *monasca* project. This is useful for providing more information about an instance.

nova_refresh

Specifies the number of seconds between calls to the nova API to refresh the instance cache. This is helpful for updating VM hostname and pruning deleted instances from the cache. By default, it is set to 14,400 seconds (four hours). Set to 0 to refresh every time the Collector runs, or to `None` to disable regular refreshes entirely. In this case, the instance cache will only be refreshed when a new instance is detected.

ping_check

Includes the entire ping command (without the IP address, which is automatically appended) to perform a ping check against instances. The `NAMESPACE` keyword is automatically replaced with the appropriate network namespace for the VM being monitored. Set to `False` to disable ping checks.

vnic_collection_period

Specifies a minimum interval in seconds for collecting disk metrics. Increase this value to reduce I/O load. If the value is lower than the global agent collection period (`check_frequency`), it will be ignored in favor of the global collection period.

vm_cpu_check_enable

Toggles the collection of VM CPU metrics. Set to `true` to enable.

vm_disks_check_enable

Toggles the collection of VM disk metrics. Set to `true` to enable.

vm_extended_disks_check_enable

Toggles the collection of extended disk metrics. Set to `true` to enable.

vm_network_check_enable

Toggles the collection of VM network metrics. Set to `true` to enable.

vm_ping_check_enable

Toggles ping checks for checking whether a host is alive. Set to `true` to enable.

vm_probation

Specifies a period of time (in seconds) in which to suspend metrics from a newly-created VM. This is to prevent quickly-obsolete metrics in an environment with a high amount of instance churn (VMs created and destroyed in rapid succession). The default probation length is 300 seconds (5 minutes). Set to 0 to disable VM probation. In this case, metrics are recorded immediately after a VM is created.

vnic_collection_period

Specifies a minimum interval in seconds for collecting VM network metrics. Increase this value to reduce I/O load. If the value is lower than the global agent collection period (`check_frequency`), it will be ignored in favor of the global collection period.

Deployment

The monasca component consists of following roles:

monasca-server

monasca server-side components that are deployed by Chef. Currently, this only creates keystone resources required by monasca, such as users, roles, endpoints, etc. The rest is left to the Ansible-based `monasca-installer` run by the `monasca-master` role.

monasca-master

Runs the Ansible-based `monasca-installer` from the Crowbar node. The installer deploys the monasca server-side components to the node that has the `monasca-server` role assigned to it. These components are `openstack-monasca-api`, and `openstack-monasca-log-api`, as well as all the back-end services they use.

monasca-agent

Deploys `openstack-monasca-agent` that is responsible for sending metrics to `monasca-api` on nodes it is assigned to.

monasca-log-agent

Deploys `openstack-monasca-log-agent` responsible for sending logs to `monasca-log-api` on nodes it is assigned to.

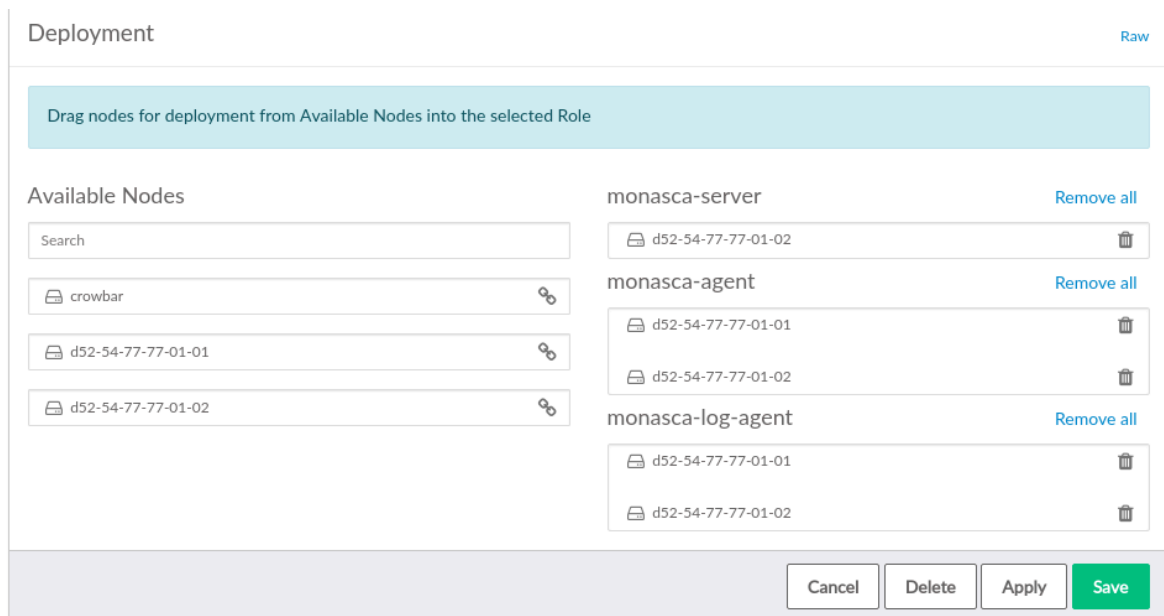


FIGURE 12.10: THE MONASCA BARCLAMP: NODE DEPLOYMENT EXAMPLE

12.7 Deploying swift (optional)

swift adds an object storage service to SUSE OpenStack Cloud for storing single files such as images or snapshots. It offers high data security by storing the data redundantly on a pool of Storage Nodes—therefore swift needs to be installed on at least two dedicated nodes.

To properly configure swift it is important to understand how it places the data. Data is always stored redundantly within the hierarchy. The swift hierarchy in SUSE OpenStack Cloud is formed out of zones, nodes, hard disks, and logical partitions. Zones are physically separated clusters, for example different server rooms each with its own power supply and network segment. A failure of one zone must not affect another zone. The next level in the hierarchy are the individual swift storage nodes (on which *swift-storage* has been deployed), followed by the hard disks. Logical partitions come last.

swift automatically places three copies of each object on the highest hierarchy level possible. If three zones are available, then each copy of the object will be placed in a different zone. In a one zone setup with more than two nodes, the object copies will each be stored on a different node. In a one zone setup with two nodes, the copies will be distributed on different hard disks. If no other hierarchy element fits, logical partitions are used.

The following attributes can be set to configure swift:

Allow Public Containers

Set to `true` to enable public access to containers.

Enable Object Versioning

If set to true, a copy of the current version is archived each time an object is updated.

Zones

Number of zones (see above). If you do not have different independent installations of storage nodes, set the number of zones to `1`.

Create 2^X Logical Partitions

Partition power. The number entered here is used to compute the number of logical partitions to be created in the cluster. The number you enter is used as a power of 2 (2^X).

We recommend using a minimum of 100 partitions per disk. To measure the partition power for your setup, multiply the number of disks from all swift nodes by 100, and then round up to the nearest power of two. Keep in mind that the first disk of each node is not used by swift, but rather for the operating system.

Example: 10 swift nodes with 5 hard disks each. Four hard disks on each node are used for swift, so there is a total of forty disks. $40 \times 100 = 4000$. The nearest power of two, 4096, equals 2^{12} . So the partition power that needs to be entered is 12.



Important: Value Cannot be Changed After the Proposal Has Been Deployed

Changing the number of logical partition after swift has been deployed is not supported. Therefore the value for the partition power should be calculated from the maximum number of partitions this cloud installation is likely going to need at any point in time.

Minimum Hours before Partition is reassigned

This option sets the number of hours before a logical partition is considered for relocation. 24 is the recommended value.

Replicas

The number of copies generated for each object. The number of replicas depends on the number of disks and zones.

Replication interval (in seconds)

Time (in seconds) after which to start a new replication process.

Debug

Shows debugging output in the log files when set to true.

SSL Support: Protocol

Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If you choose *HTTPS*, you have two options. You can either *Generate (self-signed) certificates* or provide the locations for the certificate key pair files. Using self-signed certificates is for testing purposes only and should never be used in production environments!

The screenshot shows the Swift configuration interface. At the top, there is a header with the word "Swift" and a yellow warning icon. Below the header, the page is titled "Attributes" with a "Raw" link on the right. The configuration is organized into several sections:

- Allow Public Containers (Anonymous access, performance penalty):** A dropdown menu set to "false".
- Enable Object Versioning:** A dropdown menu set to "false".
- Zones:** A text input field containing the number "2".
- Create 2^X Logical Partitions:** A text input field containing the number "16".
- Minimum Hours before Partition is reassigned:** A text input field containing the number "24".
- Replicas:** A text input field containing the number "1".
- Replication interval (in seconds):** A text input field containing the number "60".
- SSL Support:** A section header.
- Protocol:** A dropdown menu set to "HTTP".

FIGURE 12.11: THE SWIFT BARCLAMP

Apart from the general configuration described above, the swift barclamp lets you also activate and configure *Additional Middlewares*. The features these middlewares provide can be used via the swift command line client only. The Ratelimit and S3 middleware provide for the most interesting features, and we recommend enabling other middleware only for specific use-cases.

S3 Middleware

Provides an S3 compatible API on top of swift.

StaticWeb

Serve container data as a static Web site with an index file and optional file listings. See <http://docs.openstack.org/developer/swift/middleware.html#staticweb> for details.

This middleware requires setting *Allow Public Containers* to true.

TempURL

Create URLs to provide time-limited access to objects. See <http://docs.openstack.org/developer/swift/middleware.html#tempurl> for details.

FormPOST

Upload files to a container via Web form. See <http://docs.openstack.org/developer/swift/middleware.html#formpost> for details.

Bulk

Extract TAR archives into a swift account, and delete multiple objects or containers with a single request. See <http://docs.openstack.org/developer/swift/middleware.html#module-swift.common.middleware.bulk> for details.

Cross-domain

Interact with the swift API via Flash, Java, and Silverlight from an external network. See <http://docs.openstack.org/developer/swift/middleware.html#module-swift.common.middleware.crossdomain> for details.

Domain Remap

Translates container and account parts of a domain to path parameters that the swift proxy server understands. Can be used to create short URLs that are easy to remember, for example by rewriting `home.tux.example.com/$ROOT/tux/home/myfile` to `home.tux.example.com/myfile`. See http://docs.openstack.org/developer/swift/middleware.html#module-swift.common.middleware.domain_remap for details.

Ratelimit

Throttle resources such as requests per minute to provide denial of service protection. See <http://docs.openstack.org/developer/swift/middleware.html#module-swift.common.middleware.ratelimit> for details.

The swift component consists of four different roles. Deploying *swift-dispersion* is optional:

swift-storage

The virtual object storage service. Install this role on all dedicated swift Storage Nodes (at least two), but not on any other node.



Warning: swift-storage Needs Dedicated Machines

Never install the swift-storage service on a node that runs other OpenStack components.

swift-ring-compute

The ring maintains the information about the location of objects, replicas, and devices. It can be compared to an index that is used by various OpenStack components to look up the physical location of objects. *swift-ring-compute* must only be installed on a single node, preferably a Control Node.

swift-proxy

The swift proxy server takes care of routing requests to swift. Installing a single instance of *swift-proxy* on a Control Node is recommended. The *swift-proxy* role can be made highly available by deploying it on a cluster.

swift-dispersion

Deploying *swift-dispersion* is optional. The swift dispersion tools can be used to test the health of the cluster. It creates a heap of dummy objects (using 1% of the total space available). The state of these objects can be queried using the *swift-dispersion-report* query. *swift-dispersion* needs to be installed on a Control Node.

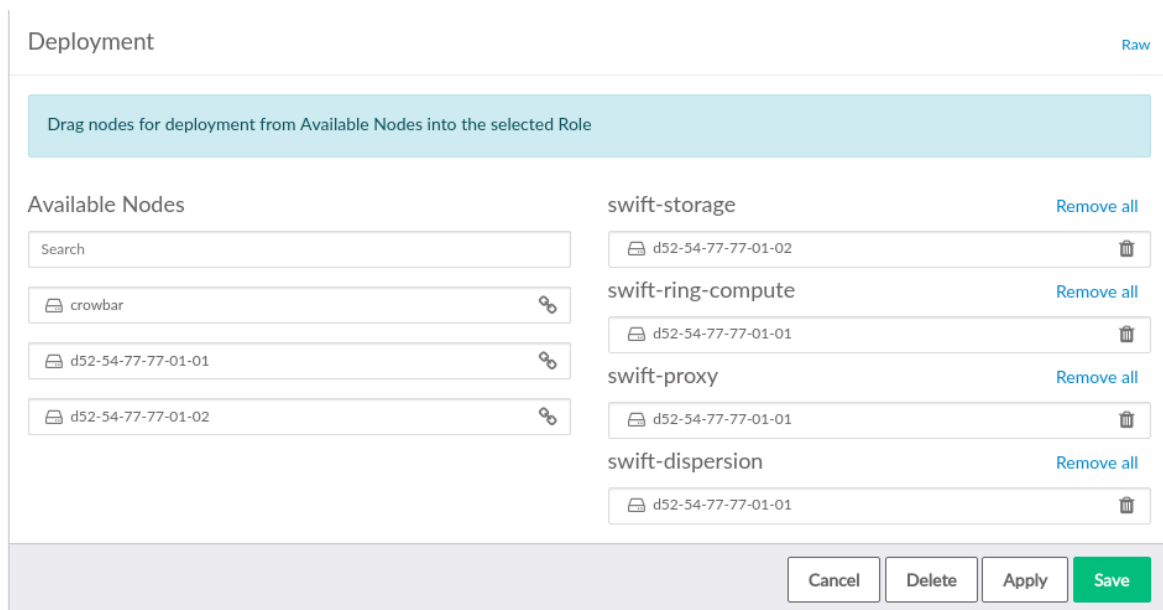


FIGURE 12.12: THE SWIFT BARCLAMP: NODE DEPLOYMENT EXAMPLE

12.7.1 HA Setup for swift

swift replicates by design, so there is no need for a special HA setup. Make sure to fulfill the requirements listed in [Section 2.6.4.1, “swift—Avoiding Points of Failure”](#).

12.8 Deploying glance

glance provides discovery, registration, and delivery services for virtual disk images. An image is needed to start an instance—it is its pre-installed root-partition. All images you want to use in your cloud to boot instances from, are provided by glance. glance must be deployed onto a Control Node. glance can be made highly available by deploying it on a cluster.

There are a lot of options to configure glance. The most important ones are explained below—for a complete reference refer to <https://github.com/crowbar/crowbar-openstack/blob/master/glance.yml>.



Important: glance API Versions

As of SUSE OpenStack Cloud Crowbar 7, the glance API v1 is no longer enabled by default. Instead, glance API v2 is used by default.

If you need to re-enable API v1 for compatibility reasons:

1. Switch to the *Raw* view of the glance barclamp.
2. Search for the `enable_v1` entry and set it to `true`:

```
"enable_v1": true
```

In new installations, this entry is set to `false` by default. When upgrading from an older version of SUSE OpenStack Cloud Crowbar it is set to `true` by default.

3. Apply your changes.

Image Storage: Default Storage Store

File. Images are stored in an image file on the Control Node.

cinder. Provides volume block storage to SUSE OpenStack Cloud Crowbar. Use it to store images.

swift. Provides an object storage service to SUSE OpenStack Cloud Crowbar.

Rados. SUSE Enterprise Storage (based on Ceph) provides block storage service to SUSE OpenStack Cloud Crowbar.

VMware. If you are using VMware as a hypervisor, it is recommended to use *VMware* for storing images. This will make starting VMware instances much faster.

Expose Backend Store Location. If this is set to *true*, the API will communicate the direct URL of the image's back-end location to HTTP clients. Set to *false* by default.

Depending on the storage back-end, there are additional configuration options available:

File Store Parameters

Only required if *Default Storage Store* is set to *File*.

Image Store Directory

Specify the directory to host the image file. The directory specified here can also be an NFS share. See [Section 11.4.3, "Mounting NFS Shares on a Node"](#) for more information.

swift Store Parameters

Only required if *Default Storage Store* is set to *swift*.

swift Container

Set the name of the container to use for the images in swift.

RADOS Store Parameters

Only required if *Default Storage Store* is set to *Rados*.

RADOS User for CephX Authentication

If you are using an external Ceph cluster, specify the user you have set up for glance (see [Section 11.4.4, "Using an Externally Managed Ceph Cluster"](#) for more information).

RADOS Pool for glance images

If you are using a SUSE OpenStack Cloud internal Ceph setup, the pool you specify here is created if it does not exist. If you are using an external Ceph cluster, specify the pool you have set up for glance (see [Section 11.4.4, "Using an Externally Managed Ceph Cluster"](#) for more information).

VMware Store Parameters

Only required if *Default Storage Store* is set to *VMware*.

vCenter Host/IP Address

Name or IP address of the vCenter server.

vCenter Username / vCenter Password

vCenter login credentials.

Datstores for Storing Images

A comma-separated list of datstores specified in the format: DATACENTER_NAME : DATASTORE_NAME

Path on the datastore, where the glance images will be stored

Specify an absolute path here.

SSL Support: Protocol

Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If you choose *HTTPS*, refer to *SSL Support: Protocol* for configuration details.

Caching

Enable and configure image caching in this section. By default, image caching is disabled. You can see this the Raw view of your nova barclamp:

```
image_cache_manager_interval = -1
```

This option sets the number of seconds to wait between runs of the image cache manager. Disabling it means that the cache manager will not automatically remove the unused images from the cache, so if you have many glance images and are running out of storage you must manually remove the unused images from the cache. We recommend leaving this option disabled as it is known to cause issues, especially with shared storage. The cache manager may remove images still in use, e.g. when network outages cause synchronization problems with compute nodes.

If you wish to enable caching, re-enable it in a custom nova configuration file, for example `/etc/nova/nova.conf.d/500-nova.conf`. This sets the interval to four minutes:

```
image_cache_manager_interval = 2400
```

See *Chapter 14, Configuration Files for OpenStack Services* for more information on custom configurations.

Learn more about glance's caching feature at <http://docs.openstack.org/developer/glance/cache.html>.

Logging: Verbose Logging

Shows debugging output in the log files when set to *true*.

Glance !

Attributes Raw

Image Storage

Default Storage Store

Expose Backend Store Location

File Store Parameters

Image Store Directory

Swift Store Parameters

Swift Container

RADOS Store Parameters

Use SES Configuration

FIGURE 12.13: THE GLANCE BARCLAMP

12.8.1 HA Setup for glance

glance can be made highly available by deploying it on a cluster. We strongly recommended doing this for the image data as well. The recommended way is to use swift or an external Ceph cluster for the image repository. If you are using a directory on the node instead (file storage back-end), you should set up shared storage on the cluster for it.

12.9 Deploying cinder

cinder, the successor of `nova-volume`, provides volume block storage. It adds persistent storage to an instance that persists until deleted, contrary to ephemeral volumes that only persist while the instance is running.

cinder can provide volume storage by using different back-ends such as local file, one or more local disks, Ceph (RADOS), VMware, or network storage solutions from EMC, EqualLogic, Fujitsu, NetApp or Pure Storage. Since SUSE OpenStack Cloud Crowbar 5, cinder supports using several back-ends simultaneously. It is also possible to deploy the same network storage back-end multiple times and therefore use different installations at the same time.

The attributes that can be set to configure cinder depend on the back-end. The only general option is *SSL Support: Protocol* (see *SSL Support: Protocol* for configuration details).



Tip: Adding or Changing a Back-End

When first opening the cinder barclamp, the default proposal—*Raw Devices*—is already available for configuration. To optionally add a back-end, go to the section *Add New cinder Back-End* and choose a *Type Of Volume* from the drop-down box. Optionally, specify the *Name for the Backend*. This is recommended when deploying the same volume type more than once. Existing back-end configurations (including the default one) can be deleted by clicking the trashcan icon if no longer needed. Note that you must configure at least one back-end.

Raw devices (local disks)

Disk Selection Method

Choose whether to use the *First Available* disk or *All Available* disks. “Available disks” are all disks currently not used by the system. Note that one disk (usually `/dev/sda`) of every block storage node is already used for the operating system and is not available for cinder.

Name of Volume

Specify a name for the cinder volume.

EMC (EMC² Storage)

IP address of the ECOM server / Port of the ECOM server

IP address and Port of the ECOM server.

Username for accessing the ECOM server / Password for accessing the ECOM server

Login credentials for the ECOM server.

VMAX port groups to expose volumes managed by this backend

VMAX port groups that expose volumes managed by this back-end.

Serial number of the VMAX Array

Unique VMAX array serial number.

Pool name within a given array

Unique pool name within a given array.

FAST Policy name to be used

Name of the FAST Policy to be used. When specified, volumes managed by this back-end are managed as under FAST control.

For more information on the EMC driver refer to the OpenStack documentation at <http://docs.openstack.org/liberty/config-reference/content/emc-vmax-driver.html>.

EqualLogic

EqualLogic drivers are included as a technology preview and are not supported.

Fujitsu ETERNUS DX

Connection Protocol

Select the protocol used to connect, either *FibreChannel* or *iSCSI*.

IP for SMI-S / Port for SMI-S

IP address and port of the ETERNUS SMI-S Server.

Username for SMI-S / Password for SMI-S

Login credentials for the ETERNUS SMI-S Server.

Snapshot (Thick/RAID Group) Pool Name

Storage pool (RAID group) in which the volumes are created. Make sure that the RAID group on the server has already been created. If a RAID group that does not exist is specified, the RAID group is built from unused disk drives. The RAID level is automatically determined by the ETERNUS DX Disk storage system.

Hitachi HUSVM

For information on configuring the Hitachi HUSVM back-end, refer to <http://docs.openstack.org/ocata/config-reference/block-storage/drivers/hitachi-storage-volume-driver.html>.

NetApp

Storage Family Type / Storage Protocol

SUSE OpenStack Cloud can use “Data ONTAP” in *7-Mode*, or in *Clustered Mode*. In *7-Mode* vFiler will be configured, in *Clustered Mode* vServer will be configured. The *Storage Protocol* can be set to either *iSCSI* or *NFS*. Choose the driver and the protocol your NetApp is licensed for.

Server host name

The management IP address for the 7-Mode storage controller, or the cluster management IP address for the clustered Data ONTAP.

Transport Type

Transport protocol for communicating with the storage controller or clustered Data ONTAP. Supported protocols are HTTP and HTTPS. Choose the protocol your NetApp is licensed for.

Server port

The port to use for communication. Port 80 is usually used for HTTP, 443 for HTTPS.

Username for accessing NetApp / Password for Accessing NetApp

Login credentials.

The vFiler Unit Name for provisioning OpenStack volumes (netapp_vfiler)

The vFiler unit to be used for provisioning of OpenStack volumes. This setting is only available in *7-Mode*.

Restrict provisioning on iSCSI to these volumes (netapp_volume_list)

Provide a list of comma-separated volume names to be used for provisioning. This setting is only available when using iSCSI as storage protocol.

NFS

List of NFS Exports

A list of available file systems on an NFS server. Enter your NFS mountpoints in the *List of NFS Exports* form in this format: `host:mountpoint -o options`. For example:

```
host1:/srv/nfs/share1 /mnt/nfs/share1 -o rsize=8192,wsiz=8192,timeo=14,intr
```

Pure Storage (FlashArray)

IP address of the management VIP

IP address of the FlashArray management VIP

API token for the FlashArray

API token for access to the FlashArray

iSCSI CHAP authentication enabled

Enable or disable iSCSI CHAP authentication

For more information on the Pure Storage FlashArray driver refer to the OpenStack documentation at <https://docs.openstack.org/ocata/config-reference/block-storage/drivers/pure-storage-driver.html>.

RADOS (Ceph)

Use Ceph Deployed by Crowbar

Select *false*, if you are using an external Ceph cluster (see *Section 11.4.4, "Using an Externally Managed Ceph Cluster"* for setup instructions).

RADOS pool for cinder volumes

Name of the pool used to store the cinder volumes.

RADOS user (Set Only if Using CephX authentication)

Ceph user name.

VMware Parameters

vCenter Host/IP Address

Host name or IP address of the vCenter server.

vCenter Username / vCenter Password

vCenter login credentials.

vCenter Cluster Names for Volumes

Provide a comma-separated list of cluster names.

Folder for Volumes

Path to the directory used to store the cinder volumes.

CA file for verifying the vCenter certificate

Absolute path to the vCenter CA certificate.

vCenter SSL Certificate is insecure (for instance, self-signed)

Default value: `false` (the CA truststore is used for verification). Set this option to `true` when using self-signed certificates to disable certificate checks. This setting is for testing purposes only and must not be used in production environments!

Local file

Volume File Name

Absolute path to the file to be used for block storage.

Maximum File Size (GB)

Maximum size of the volume file. Make sure not to overcommit the size, since it will result in data loss.

Name of Volume

Specify a name for the cinder volume.

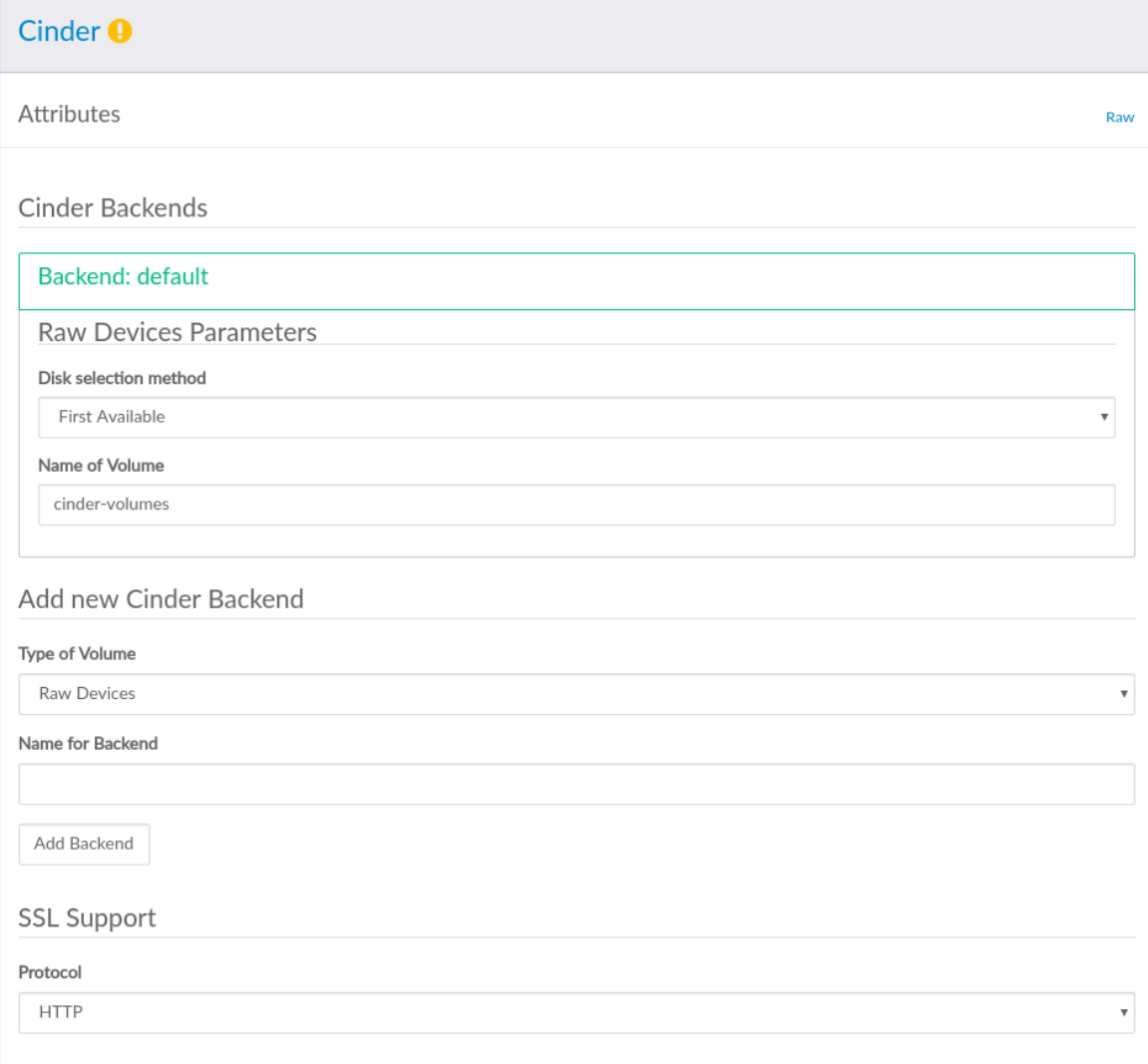


Note: Using *Local File* for Block Storage

Using a file for block storage is not recommended for production systems, because of performance and data security reasons.

Other driver

Lets you manually pick and configure a driver. Only use this option for testing purposes, as it is not supported.



The screenshot shows the Cinder Barclamp configuration interface. At the top, there is a header with the word "Cinder" and a warning icon. Below the header, there is a section for "Attributes" with a "Raw" link. The main section is titled "Cinder Backends" and contains a table with one entry: "Backend: default". Below this table, there is a section for "Raw Devices Parameters" with two dropdown menus: "Disk selection method" (set to "First Available") and "Name of Volume" (set to "cinder-volumes"). Below this, there is a section for "Add new Cinder Backend" with a "Type of Volume" dropdown (set to "Raw Devices"), a "Name for Backend" text input field, and an "Add Backend" button. At the bottom, there is a section for "SSL Support" with a "Protocol" dropdown (set to "HTTP").

FIGURE 12.14: THE CINDER BARCLAMP

The cinder component consists of two different roles:

cinder-controller

The cinder controller provides the scheduler and the API. Installing *cinder-controller* on a Control Node is recommended.

cinder-volume

The virtual block storage service. It can be installed on a Control Node. However, we recommend deploying it on one or more dedicated nodes supplied with sufficient networking capacity to handle the increase in network traffic.

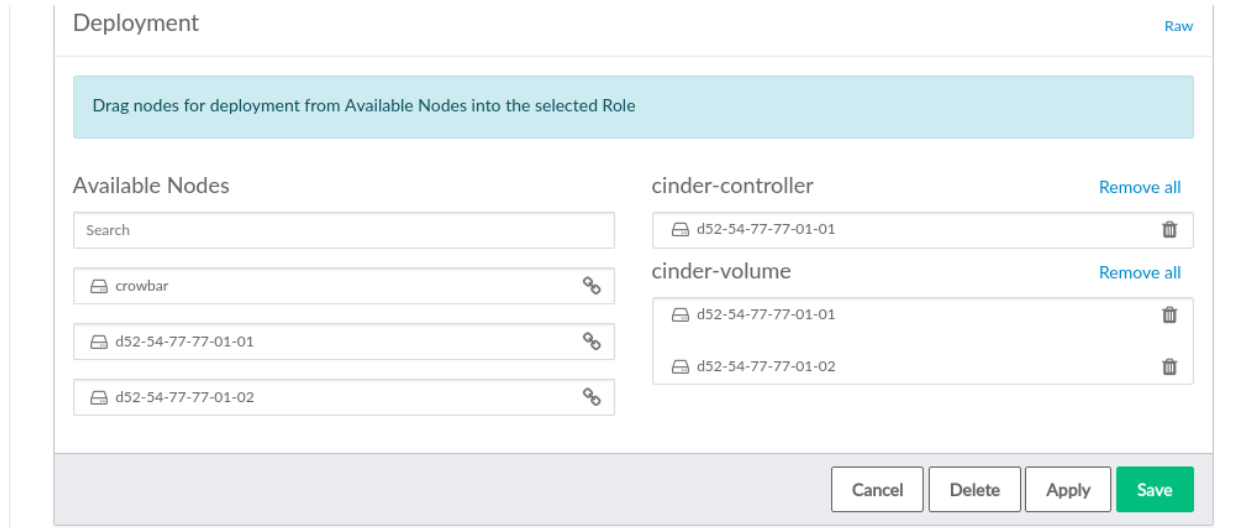


FIGURE 12.15: THE CINDER BARCLAMP: NODE DEPLOYMENT EXAMPLE

12.9.1 HA Setup for cinder

Both the *cinder-controller* and the *cinder-volume* role can be deployed on a cluster.



Note: Moving *cinder-volume* to a Cluster

If you need to re-deploy *cinder-volume* role from a single machine to a cluster environment, the following will happen: Volumes that are currently attached to instances will continue to work, but adding volumes to instances will not succeed.

To solve this issue, run the following script once on each node that belongs to the *cinder-volume* cluster: `/usr/bin/cinder-migrate-volume-names-to-cluster`.

The script is automatically installed by Crowbar on every machine or cluster that has a *cinder-volume* role applied to it.

In combination with Ceph or a network storage solution, deploying cinder in a cluster minimizes the potential downtime. For *cinder-volume* to be applicable to a cluster, the role needs all cinder backends to be configured for non-local storage. If you are using local volumes or raw devices in any of your volume backends, you cannot apply *cinder-volume* to a cluster.

12.10 Deploying neutron

neutron provides network connectivity between interface devices managed by other OpenStack components (most likely nova). The service works by enabling users to create their own networks and then attach interfaces to them.

neutron must be deployed on a Control Node. You first need to choose a core plug-in—*ml2* or *vmware*. Depending on your choice, more configuration options will become available.

The *vmware* option lets you use an existing VMware NSX installation. Using this plugin is not a prerequisite for the VMware vSphere hypervisor support. However, it is needed when wanting to have security groups supported on VMware compute nodes. For all other scenarios, choose *ml2*. The only global option that can be configured is *SSL Support*. Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If choosing *HTTPS*, refer to [SSL Support: Protocol](#) for configuration details.

ml2 (Modular Layer 2)

Modular Layer 2 Mechanism Drivers

Select which mechanism driver(s) shall be enabled for the *ml2* plugin. It is possible to select more than one driver by holding the **Ctrl** key while clicking. Choices are:

openvswitch. Supports GRE, VLAN and VXLAN networks (to be configured via the *Modular Layer 2 type drivers* setting). VXLAN is the default.

linuxbridge. Supports VLANs only. Requires to specify the *Maximum Number of VLANs*.

cisco_nexus. Enables neutron to dynamically adjust the VLAN settings of the ports of an existing Cisco Nexus switch when instances are launched. It also requires *openvswitch* which will automatically be selected. With *Modular Layer 2 type drivers*, *vlan* must be added. This option also requires to specify the *Cisco Switch Credentials*. See [Appendix A, Using Cisco Nexus Switches with neutron](#) for details.

vmware_dvs. *vmware_dvs* driver makes it possible to use neutron for networking in a VMware-based environment. Choosing *vmware_dvs*, automatically selects the required *openswitch*, *vxlan*, and *vlan* drivers. In the *Raw* view, it is also possible to configure two additional attributes: *clean_on_start* (clean up the DVS portgroups on the target vCenter Servers when neutron-server is restarted) and *precreate_networks* (create DVS portgroups corresponding to networks in advance, rather than when virtual machines are attached to these networks).

Use Distributed Virtual Router Setup

With the default setup, all intra-Compute Node traffic flows through the network Control Node. The same is true for all traffic from floating IPs. In large deployments the network Control Node can therefore quickly become a bottleneck. When this option is set to *true*, network agents will be installed on all compute nodes. This will de-centralize the network traffic, since Compute Nodes will be able to directly “talk” to each other. Distributed Virtual Routers (DVR) require the *openswitch* driver and will not work with the *linuxbridge* driver. For details on DVR refer to <https://wiki.openstack.org/wiki/Neutron/DVR>.

Modular Layer 2 Type Drivers

This option is only available when having chosen the *openswitch* or the *cisco_nexus* mechanism drivers. Options are *vlan*, *gre* and *vxlan*. It is possible to select more than one driver by holding the **Ctrl** key while clicking.

When multiple type drivers are enabled, you need to select the *Default Type Driver for Provider Network*, that will be used for newly created provider networks. This also includes the *nova_fixed* network, that will be created when applying the neutron proposal. When manually creating provider networks with the **neutron** command, the default can be overwritten with the `--provider:network_type type` switch. You will also need to set a *Default Type Driver for Tenant Network*. It is not possible to change this default when manually creating tenant networks with the **neutron** command. The non-default type driver will only be used as a fallback.

Depending on your choice of the type driver, more configuration options become available.

gre. Having chosen *gre*, you also need to specify the start and end of the tunnel ID range.

vlan. The option *vlan* requires you to specify the *Maximum number of VLANs*.

vxlan. Having chosen *vxlan*, you also need to specify the start and end of the VNI range.



Important: Drivers for the VMware Compute Node

neutron must not be deployed with the openvswitch with gre plug-in.

z/VM Configuration

xCAT Host/IP Address

Host name or IP address of the xCAT Management Node.

xCAT Username/Password

xCAT login credentials.

rdev list for physnet1 vswitch uplink (if available)

List of rdev addresses that should be connected to this vswitch.

xCAT IP Address on Management Network

IP address of the xCAT management interface.

Net Mask of Management Network

Net mask of the xCAT management interface.

vmware

This plug-in requires to configure access to the VMware NSX service.

VMware NSX User Name/Password

Login credentials for the VMware NSX server. The user needs to have administrator permissions on the NSX server.

VMware NSX Controllers

Enter the IP address and the port number (IP-ADDRESS : PORT) of the controller API endpoint. If the port number is omitted, port 443 will be used. You may also enter multiple API endpoints (comma-separated), provided they all belong to the same controller cluster. When multiple API endpoints are specified, the plugin will load balance requests on the various API endpoints.

UUID of the NSX Transport Zone/Gateway Service

The UUIDs for the transport zone and the gateway service can be obtained from the NSX server. They will be used when networks are created.

Neutron !

Attributes Raw

DNS Domain

openstack.local

Networking Plugin

Plugin

ml2

Modular Layer 2 mechanism drivers

- openvswitch
- linuxbridge
- cisco_nexus
- vmware_dvs
- cisco_apic_ml2
- apic_gbp

Modular Layer 2 type drivers

- vxlan
- vlan
- gre
- opflex

Use L2 Population

false

Use Distributed Virtual Router setup

false

FIGURE 12.16: THE NEUTRON BARCLAMP

The neutron component consists of two different roles:

neutron-server

neutron-server provides the scheduler and the API. It needs to be installed on a Control Node.

neutron-network

This service runs the various agents that manage the network traffic of all the cloud instances. It acts as the DHCP and DNS server and as a gateway for all cloud instances. It is recommend to deploy this role on a dedicated node supplied with sufficient network capacity.

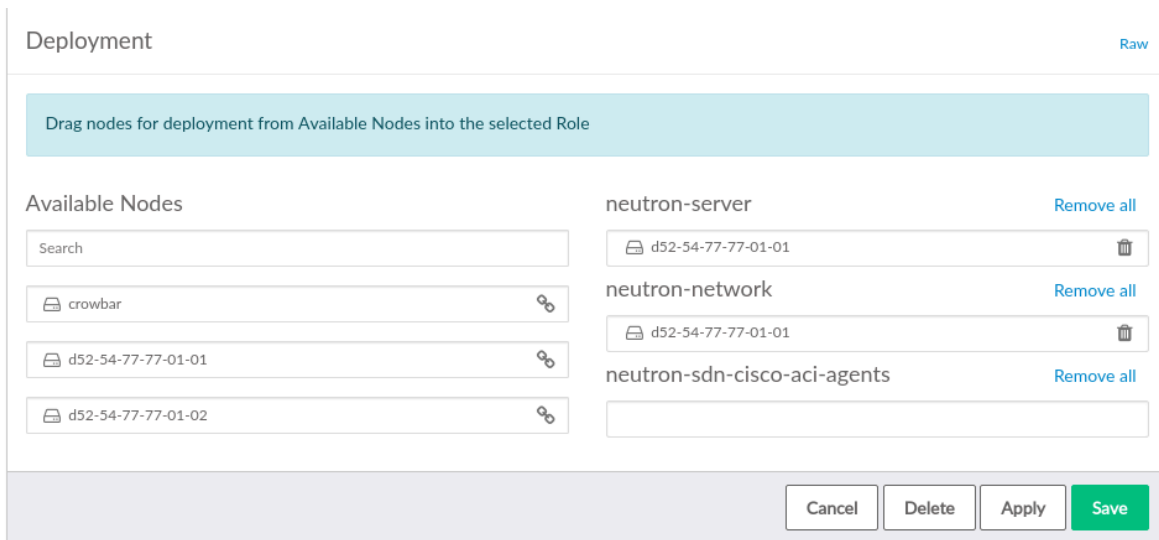


FIGURE 12.17: THE NEUTRON BARCLAMP

12.10.1 Using Infoblox IPAM Plug-in

In the neutron barclamp, you can enable support for the infoblox IPAM plug-in and configure it. For configuration, the `infoblox` section contains the subsections `grids` and `grid_defaults`.

grids

This subsection must contain at least one entry. For each entry, the following parameters are required:

- `admin_user_name`
- `admin_password`
- `grid_master_host`
- `grid_master_name`
- `data_center_name`

You can also add multiple entries to the `grids` section. However, the upstream infoblox agent only supports a single grid currently.

grid_defaults

This subsection contains the default settings that are used for each grid (unless you have configured specific settings within the `grids` section).

For detailed information on all infoblox-related configuration settings, see <https://github.com/openstack/networking-infoblox/blob/master/doc/source/installation.rst>.

Currently, all configuration options for infoblox are only available in the raw mode of the neutron barclamp. To enable support for the infoblox IPAM plug-in and configure it, proceed as follows:

1. *Edit* the neutron barclamp proposal or create a new one.

2. Click *Raw* and search for the following section:

```
"use_infoblox": false,
```

3. To enable support for the infoblox IPAM plug-in, change this entry to:

```
"use_infoblox": true,
```

4. In the grids section, configure at least one grid by replacing the example values for each parameter with real values.

5. If you need specific settings for a grid, add some of the parameters from the grid_defaults section to the respective grid entry and adjust their values.

Otherwise Crowbar applies the default setting to each grid when you save the barclamp proposal.

6. Save your changes and apply them.

12.10.2 HA Setup for neutron

neutron can be made highly available by deploying *neutron-server* and *neutron-network* on a cluster. While *neutron-server* may be deployed on a cluster shared with other services, it is strongly recommended to use a dedicated cluster solely for the *neutron-network* role.

12.10.3 Setting Up Multiple External Networks

This section shows you how to create external networks on SUSE OpenStack Cloud.

12.10.3.1 New Network Configurations

1. If you have not yet deployed Crowbar, add the following configuration to `/etc/crowbar/network.json` to set up an external network, using the name of your new network, VLAN ID, and network addresses. If you have already deployed Crowbar, then add this configuration to the *Raw* view of the Network Barclamp.

```
"public2": {
  "conduit": "intf1",
  "vlan": 600,
  "use_vlan": true,
  "add_bridge": false,
  "subnet": "192.168.135.128",
  "netmask": "255.255.255.128",
  "broadcast": "192.168.135.255",
  "ranges": {
    "host": { "start": "192.168.135.129",
              "end": "192.168.135.254" }
  }
},
```

2. Modify the `additional_external_networks` in the *Raw* view of the neutron Barclamp with the name of your new external network.
3. Apply both barclamps, and it may also be necessary to re-apply the nova Barclamp.
4. Then follow the steps in the next section to create the new external network.

12.10.3.2 Create the New External Network

The following steps add the network settings, including IP address pools, gateway, routing, and virtual switches to your new network.

1. Set up interface mapping using either Open vSwitch (OVS) or Linuxbridge. For Open vSwitch run the following command:

```
openstack network create public2 --provider:network_type flat \
  --provider:physical_network public2 --router:external=True
```

For Linuxbridge run the following command:

```
openstack network create --router:external True --provider:physical_network
physnet1 \
```

```
--provider:network_type vlan --provider:segmentation_id 600
```

2. If a different network is used then Crowbar will create a new interface mapping. Then you can use a flat network:

```
openstack network create public2 --provider:network_type flat \  
--provider:physical_network public2 --router:external=True
```

3. Create a subnet:

```
openstack subnet create --name public2 --allocation-pool \  
start=192.168.135.2,end=192.168.135.127 --gateway 192.168.135.1 public2 \  
192.168.135.0/24 --enable_dhcp False
```

4. Create a router, *router2*:

```
openstack router create router2
```

5. Connect *router2* to the new external network:

```
openstack router set router2 public2
```

6. Create a new private network and connect it to *router2*

```
openstack network create priv-net  
openstack subnet create priv-net --gateway 10.10.10.1 10.10.10.0/24 \  
--name priv-net-sub  
openstack router add subnet router2 priv-net-sub
```

7. Boot a VM on *priv-net-sub* and set a security group that allows SSH.
8. Assign a floating IP address to the VM, this time from network *public2*.
9. From the node verify that SSH is working by opening an SSH session to the VM.

12.10.3.3 How the Network Bridges are Created

For OVS, a new bridge will be created by Crowbar, in this case *br-public2*. In the bridge mapping the new network will be assigned to the bridge. The interface specified in */etc/crowbar/network.json* (in this case *eth0.600*) will be plugged into *br-public2*. The new public network can be created in neutron using the new public network name as *provider:physical_network*.

For Linuxbridge, Crowbar will check the interface associated with `public2`. If this is the same as `physnet1` no interface mapping will be created. The new public network can be created in neutron using `physnet1` as physical network and specifying the correct VLAN ID:

```
openstack network create public2 --router:external True \  
  --provider:physical_network physnet1 --provider:network_type vlan \  
  --provider:segmentation_id 600
```

A bridge named `brq-NET_ID` will be created and the interface specified in `/etc/crowbar/network.json` will be plugged into it. If a new interface is associated in `/etc/crowbar/network.json` with `public2` then Crowbar will add a new interface mapping and the second public network can be created using `public2` as the physical network:

```
openstack network create public2 --provider:network_type flat \  
  --provider:physical_network public2 --router:external=True
```

12.11 Deploying nova

nova provides key services for managing the SUSE OpenStack Cloud, sets up the Compute Nodes. SUSE OpenStack Cloud currently supports KVM and VMware vSphere. The unsupported QEMU option is included to enable test setups with virtualized nodes. The following attributes can be configured for nova:

Scheduler Options: Virtual RAM to Physical RAM allocation ratio

Set the “overcommit ratio” for RAM for instances on the Compute Nodes. A ratio of `1.0` means no overcommitment. Changing this value is not recommended.

Scheduler Options: Virtual CPU to Physical CPU allocation ratio

Set the “overcommit ratio” for CPUs for instances on the Compute Nodes. A ratio of `1.0` means no overcommitment.

Scheduler Options: Virtual Disk to Physical Disk allocation ratio

Set the “overcommit ratio” for virtual disks for instances on the Compute Nodes. A ratio of `1.0` means no overcommitment.

Scheduler Options: Reserved Memory for nova-compute hosts (MB)

Amount of reserved host memory that is not used for allocating VMs by `nova-compute`.

Live Migration Support: Enable Libvirt Migration

Allows to move KVM instances to a different Compute Node running the same hypervisor (cross hypervisor migrations are not supported). Useful when a Compute Node needs to be shut down or rebooted for maintenance or when the load of the Compute Node is very high. Instances can be moved while running (Live Migration).



Warning: Libvirt Migration and Security

Enabling the libvirt migration option will open a TCP port on the Compute Nodes that allows access to all instances from all machines in the admin network. Ensure that only authorized machines have access to the admin network when enabling this option.



Tip: Specifying Network for Live Migration

It is possible to change a network to live migrate images. This is done in the raw view of the nova barclamp. In the `migration` section, change the `network` attribute to the appropriate value (for example, `storage` for Ceph).

KVM Options: Enable Kernel Samepage Merging

Kernel SamePage Merging (KSM) is a Linux Kernel feature which merges identical memory pages from multiple running processes into one memory region. Enabling it optimizes memory usage on the Compute Nodes when using the KVM hypervisor at the cost of slightly increasing CPU usage.

SSL Support: Protocol

Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If choosing *HTTPS*, refer to *SSL Support: Protocol* for configuration details.

VNC Settings: NoVNC Protocol

After having started an instance you can display its VNC console in the OpenStack Dashboard (horizon) via the browser using the noVNC implementation. By default this connection is not encrypted and can potentially be eavesdropped.

Enable encrypted communication for noVNC by choosing *HTTPS* and providing the locations for the certificate key pair files.

Logging: Verbose Logging

Shows debugging output in the log files when set to *true*.



Note: Custom Vendor Data for Instances

You can pass custom vendor data to all VMs via nova's metadata server. For example, information about a custom SMT server can be used by the SUSE guest images to automatically configure the repositories for the guest.

1. To pass custom vendor data, switch to the *Raw* view of the nova barclamp.
2. Search for the following section:

```
"metadata": {
  "vendordata": {
    "json": "{}"
  }
}
```

3. As value of the `json` entry, enter valid JSON data. For example:

```
"metadata": {
  "vendordata": {
    "json": "{\"CUSTOM_KEY\": \"CUSTOM_VALUE\"}"
  }
}
```

The string needs to be escaped because the barclamp file is in JSON format, too.

Use the following command to access the custom vendor data from inside a VM:

```
curl -s http://METADATA_SERVER/openstack/latest/vendor_data.json
```

The IP address of the metadata server is always the same from within a VM. For more details, see <https://www.suse.com/communities/blog/vms-get-access-metadata-neutron/>.

FIGURE 12.18: THE NOVA BARCLAMP

The nova component consists of eight different roles:

nova-controller

Distributing and scheduling the instances is managed by the *nova-controller*. It also provides networking and messaging services. *nova-controller* needs to be installed on a Control Node.

nova-compute-kvm / nova-compute-qemu / nova-compute-vmware /

Provides the hypervisors (KVM, QEMU, VMware vSphere, and z/VM) and tools needed to manage the instances. Only one hypervisor can be deployed on a single compute node. To use different hypervisors in your cloud, deploy different hypervisors to different Compute Nodes. A *nova-compute-** role needs to be installed on every Compute Node. However, not all hypervisors need to be deployed.

Each image that will be made available in SUSE OpenStack Cloud to start an instance is bound to a hypervisor. Each hypervisor can be deployed on multiple Compute Nodes (except for the VMware vSphere role, see below). In a multi-hypervisor deployment you should make sure to deploy the *nova-compute-** roles in a way, that enough compute power is available for each hypervisor.



Note: Re-assigning Hypervisors

Existing `nova-compute-*` nodes can be changed in a production SUSE OpenStack Cloud without service interruption. You need to “evacuate” the node, re-assign a new `nova-compute` role via the nova barclamp and *Apply* the change. `nova-compute-vmware` can only be deployed on a single node.

Deployment Raw

Drag nodes for deployment from Available Nodes into the selected Role

Available Nodes

Search

crowbar

d52-54-77-77-01-01

d52-54-77-77-01-02

nova-controller Remove all

d52-54-77-77-01-01

ec2-api Remove all

nova-compute-ironic Remove all

nova-compute-kvm Remove all

d52-54-77-77-01-02

nova-compute-qemu Remove all

nova-compute-vmware Remove all

nova-compute-zvm Remove all

Cancel Delete Apply Save

FIGURE 12.19: THE NOVA BARCLAMP: NODE DEPLOYMENT EXAMPLE WITH TWO KVM NODES

When deploying a `nova-compute-vmware` node with the `vmware_dvs` ML2 driver enabled in the neutron barclamp, the following new attributes are also available in the `vcenter` section of the *Raw* mode: `dvs_name` (the name of the DVS switch configured on the target vCenter cluster) and `dvs_security_groups` (enable or disable implementing security groups through DVS traffic rules).

It is important to specify the correct `dvs_name` value, as the barclamp expects the DVS switch to be preconfigured on the target VMware vCenter cluster.



Warning: `vmware_dvs` must be enabled

Deploying `nova-compute-vmware` nodes will not result in a functional cloud setup if the `vmware_dvs` ML2 plugin is not enabled in the neutron barclamp.

12.11.1 HA Setup for nova

Making `nova-controller` highly available requires no special configuration—it is sufficient to deploy it on a cluster.

To enable High Availability for Compute Nodes, deploy the following roles to one or more clusters with remote nodes:

- `nova-compute-kvm`
- `nova-compute-qemu`
- `ec2-api`

The cluster to which you deploy the roles above can be completely independent of the one to which the role `nova-controller` is deployed.

However, the `nova-controller` and `ec2-api` roles must be deployed the same way (either *both* to a cluster or *both* to individual nodes. This is due to Crowbar design limitations.



Tip: Shared Storage

It is recommended to use shared storage for the `/var/lib/nova/instances` directory, to ensure that ephemeral disks will be preserved during recovery of VMs from failed compute nodes. Without shared storage, any ephemeral disks will be lost, and recovery will rebuild the VM from its original image.

If an external NFS server is used, enable the following option in the nova barclamp proposal: *Shared Storage for nova instances has been manually configured.*

12.12 Deploying horizon (OpenStack Dashboard)

The last component that needs to be deployed is horizon, the OpenStack Dashboard. It provides a Web interface for users to start and stop instances and for administrators to manage users, groups, roles, etc. horizon should be installed on a Control Node. To make horizon highly available, deploy it on a cluster.

The following attributes can be configured:

Session Timeout

Timeout (in minutes) after which a user is been logged out automatically. The default value is set to four hours (240 minutes).



Note: Timeouts Larger than Four Hours

Every horizon session requires a valid keystone token. These tokens also have a lifetime of four hours (14400 seconds). Setting the horizon session timeout to a value larger than 240 will therefore have no effect, and you will receive a warning when applying the barclamp.

To successfully apply a timeout larger than four hours, you first need to adjust the keystone token expiration accordingly. To do so, open the keystone barclamp in *Raw* mode and adjust the value of the key `token_expiration`. Note that the value has to be provided in *seconds*. When the change is successfully applied, you can adjust the horizon session timeout (in *minutes*). Note that extending the keystone token expiration may cause scalability issues in large and very busy SUSE OpenStack Cloud installations.

User Password Validation: Regular expression used for password validation

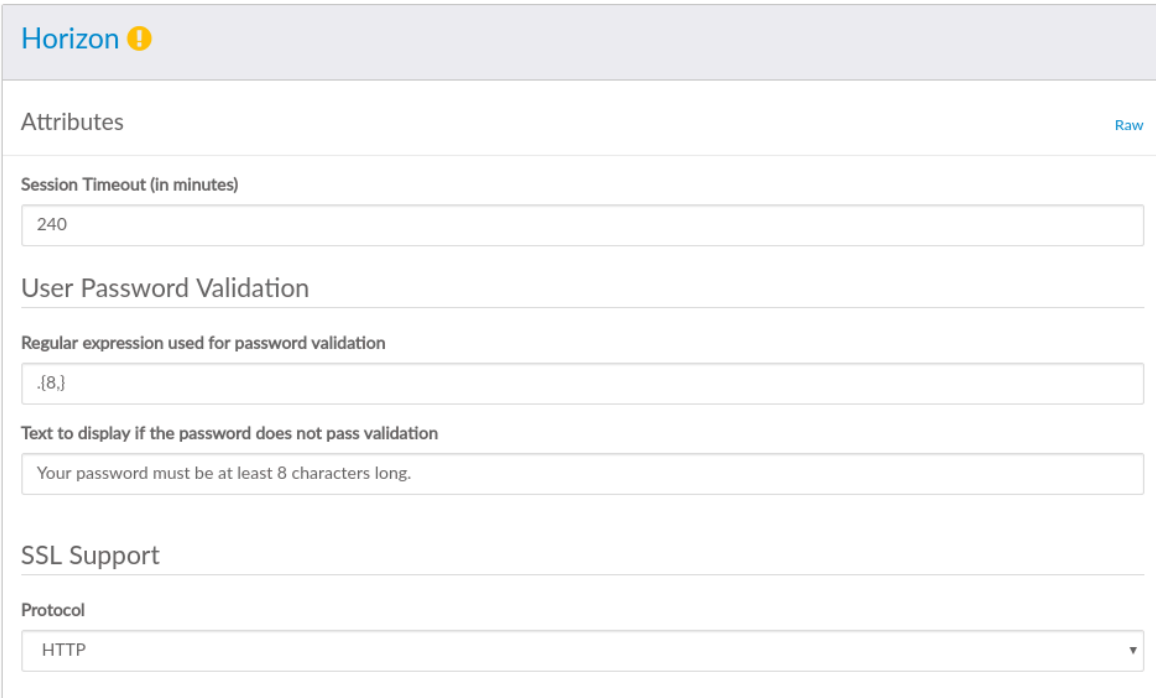
Specify a regular expression with which to check the password. The default expression (`.{8,}`) tests for a minimum length of 8 characters. The string you enter is interpreted as a Python regular expression (see <http://docs.python.org/2.7/library/re.html#module-re> for a reference).

User Password Validation: Text to display if the password does not pass validation

Error message that will be displayed in case the password validation fails.

SSL Support: Protocol

Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If choosing *HTTPS*, you have two choices. You can either *Generate (self-signed) certificates* or provide the locations for the certificate key pair files and,—optionally— the certificate chain file. Using self-signed certificates is for testing purposes only and should never be used in production environments!



The screenshot shows the Horizon configuration interface. At the top, there is a header with the word "Horizon" and a yellow warning icon. Below the header, there is a section titled "Attributes" with a "Raw" link on the right. Under "Attributes", there is a "Session Timeout (in minutes)" field with the value "240". Below that is a section titled "User Password Validation". Under this section, there is a "Regular expression used for password validation" field with the value ".{8,}". Below that is a "Text to display if the password does not pass validation" field with the value "Your password must be at least 8 characters long.". Below the "User Password Validation" section is a section titled "SSL Support". Under "SSL Support", there is a "Protocol" dropdown menu with the value "HTTP" selected.

FIGURE 12.20: THE HORIZON BARCLAMP

12.12.1 HA Setup for horizon

Making horizon highly available requires no special configuration—it is sufficient to deploy it on a cluster.

12.13 Deploying heat (Optional)

heat is a template-based orchestration engine that enables you to, for example, start workloads requiring multiple servers or to automatically restart instances if needed. It also brings auto-scaling to SUSE OpenStack Cloud by automatically starting additional instances if certain criteria are met. For more information about heat refer to the OpenStack documentation at <http://docs.openstack.org/developer/heat/>.

heat should be deployed on a Control Node. To make heat highly available, deploy it on a cluster. The following attributes can be configured for heat:

Verbose Logging

Shows debugging output in the log files when set to *true*.

SSL Support: Protocol

Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If choosing *HTTPS*, refer to *SSL Support: Protocol* for configuration details.

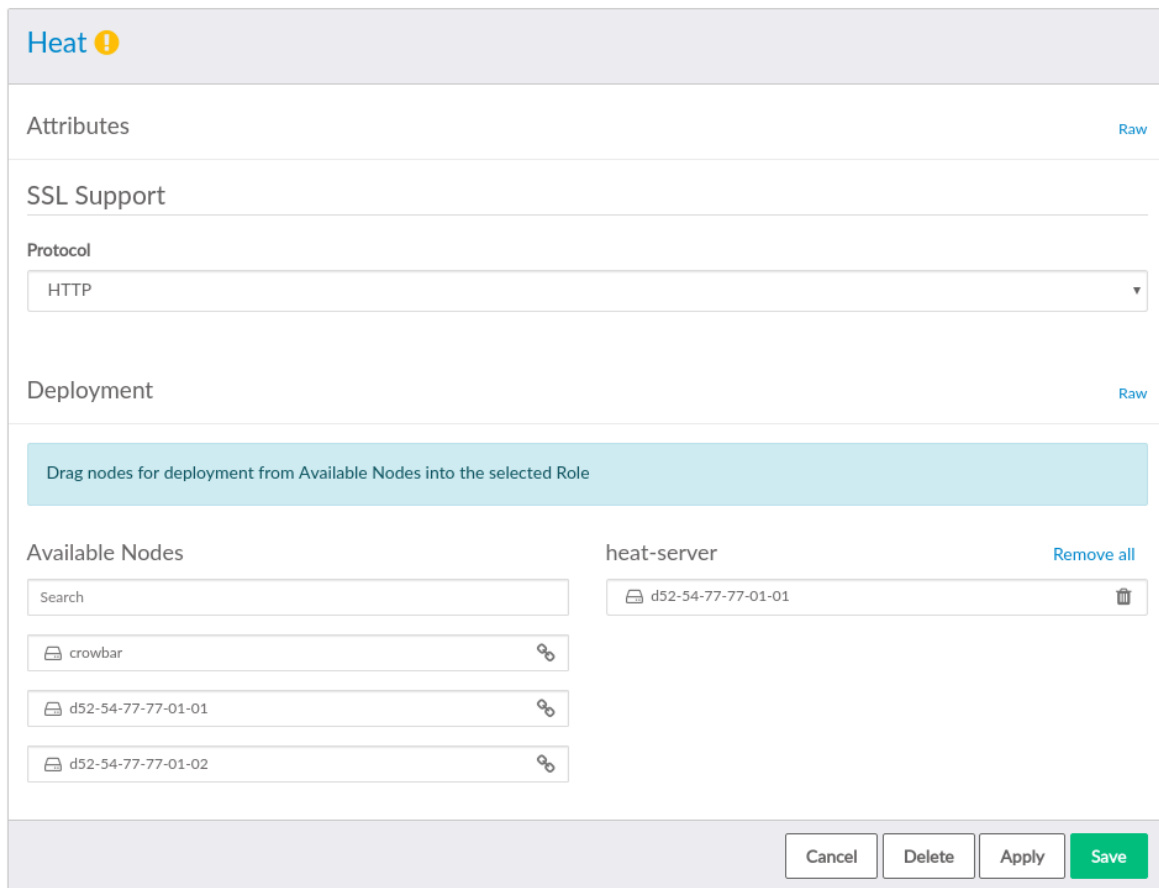


FIGURE 12.21: THE HEAT BARCLAMP

12.13.1 Enabling Identity Trusts Authorization (Optional)

heat uses keystone Trusts to delegate a subset of user roles to the heat engine for deferred operations (see [Steve Hardy's blog \(http://hardysteven.blogspot.de/2014/04/heat-auth-model-updates-part-1-trusts.html\)](http://hardysteven.blogspot.de/2014/04/heat-auth-model-updates-part-1-trusts.html) for details). It can either delegate all user roles or only those specified in the `trusts_delegated_roles` setting. Consequently, all roles listed in `trusts_delegated_roles` need to be assigned to a user, otherwise the user will not be able to use heat.

The recommended setting for `trusts_delegated_roles` is `member`, since this is the default role most users are likely to have. This is also the default setting when installing SUSE OpenStack Cloud from scratch.

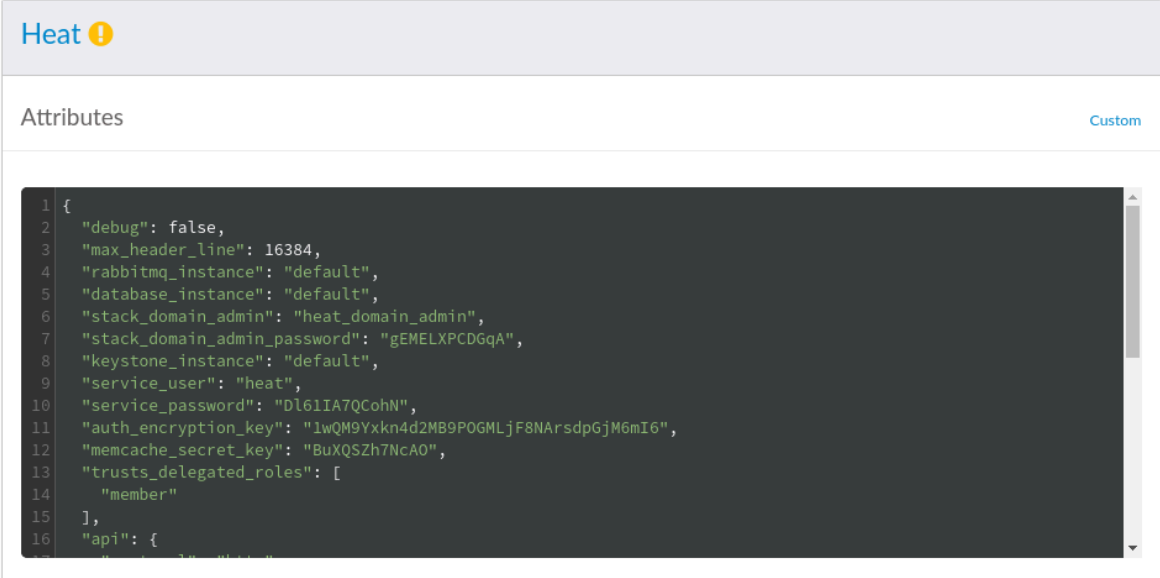
On installations where this setting is introduced through an upgrade, `trusts_delegated_roles` will be set to `heat_stack_owner`. This is a conservative choice to prevent breakage in situations where unprivileged users may already have been assigned the `heat_stack_owner`

role to enable them to use heat but lack the `member` role. As long as you can ensure that all users who have the `heat_stack_owner` role also have the `member` role, it is both safe and recommended to change `trusts_delegated_roles` to `member`.

! Important

If the Octavia barclamp is deployed, the `trusts_delegated_roles` configuration option either needs to be set to an empty value, or the `load-balancer_member` role needs to be included, otherwise it won't be possible to create Octavia load balancers via heat stacks. Refer to the [Section 12.20.3, "Migrating Users to Octavia"](#) section for more details on the list of specialized roles employed by Octavia. Also note that adding the `load-balancer_member` role to the `trusts_delegated_roles` list has the undesired side effect that only users that have this role assigned to them will be allowed to access the Heat API, as covered previously in this section.

To view or change the `trusts_delegated_role` setting you need to open the heat barclamp and click *Raw* in the *Attributes* section. Search for the `trusts_delegated_roles` setting and modify the list of roles as desired.



The screenshot shows the Heat barclamp interface in raw mode. The title bar says "Heat" with a warning icon. Below it, the "Attributes" section is visible, with a "Custom" button on the right. A code editor displays the following JSON configuration:

```
1 {
2   "debug": false,
3   "max_header_line": 16384,
4   "rabbitmq_instance": "default",
5   "database_instance": "default",
6   "stack_domain_admin": "heat_domain_admin",
7   "stack_domain_admin_password": "gEMELXPCDGqA",
8   "keystone_instance": "default",
9   "service_user": "heat",
10  "service_password": "Dl61IA7QCohN",
11  "auth_encryption_key": "1wQM9Yxkn4d2MB9POGMLjF8NArSDpGjM6mI6",
12  "memcache_secret_key": "BuXQSZh7NcA0",
13  "trusts_delegated_roles": [
14    "member"
15  ],
16  "api": {
```

FIGURE 12.22: THE HEAT BARCLAMP: RAW MODE



Warning: Empty Value

An empty value for `trusts_delegated_roles` will delegate *all* of user roles to heat. This may create a security risk for users who are assigned privileged roles, such as `admin`, because these privileged roles will also be delegated to the heat engine when these users create heat stacks.

12.13.2 HA Setup for heat

Making heat highly available requires no special configuration—it is sufficient to deploy it on a cluster.

12.14 Deploying ceilometer (Optional)

ceilometer collects CPU and networking data from SUSE OpenStack Cloud. This data can be used by a billing system to enable customer billing. Deploying ceilometer is optional. ceilometer agents use monasca database to store collected data.

For more information about ceilometer refer to the OpenStack documentation at <http://docs.openstack.org/developer/ceilometer/>.



Important: ceilometer Restrictions

As of SUSE OpenStack Cloud Crowbar 8 data measuring is only supported for KVM and Windows instances. Other hypervisors and SUSE OpenStack Cloud features such as object or block storage will not be measured.

The following attributes can be configured for ceilometer:

Intervals used for OpenStack Compute, Image, or Block Storage meter updates (in seconds)

Specify intervals in seconds after which ceilometer performs updates of specified meters.

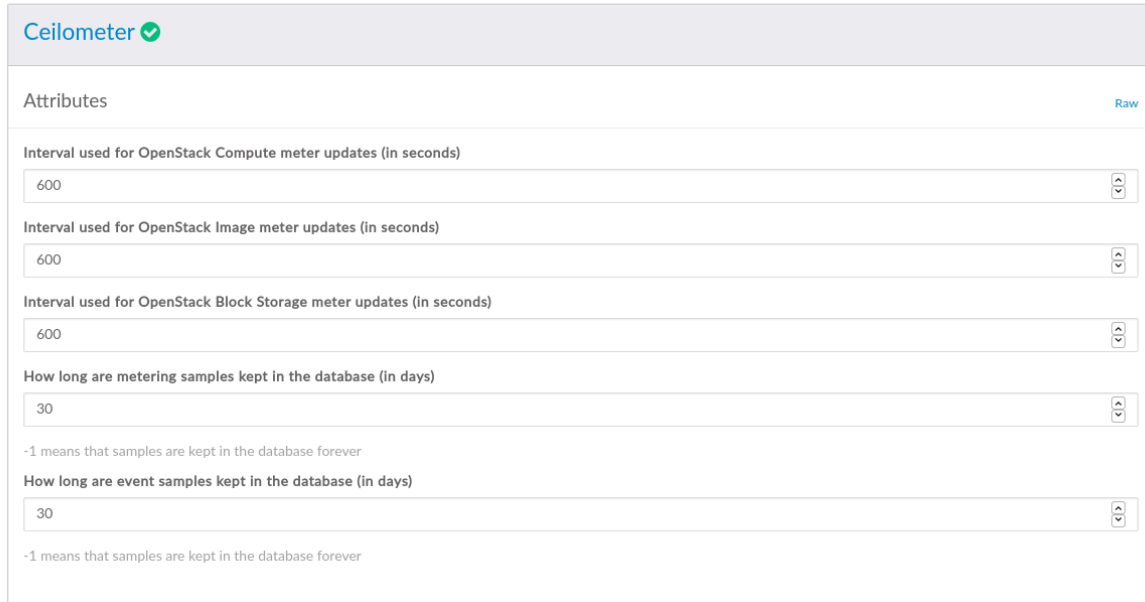
How long are metering samples kept in the database (in days)

Specify how long to keep the metering data. `-1` means that samples are kept in the database forever.

How long are event samples kept in the database (in days)

Specify how long to keep the event data. `-1` means that samples are kept in the database forever.

● Edit Proposal



The screenshot shows the 'Ceilometer' configuration page. At the top, there is a header 'Ceilometer' with a checkmark icon. Below it, the 'Attributes' section is visible, with a 'Raw' link on the right. The configuration includes several input fields with dropdown arrows:

- Interval used for OpenStack Compute meter updates (in seconds): 600
- Interval used for OpenStack Image meter updates (in seconds): 600
- Interval used for OpenStack Block Storage meter updates (in seconds): 600
- How long are metering samples kept in the database (in days): 30
- How long are event samples kept in the database (in days): 30

Below the last two fields, there is a note: '-1 means that samples are kept in the database forever'.

FIGURE 12.23: THE CEILOMETER BARCLAMP

The ceilometer component consists of four different roles:

ceilometer-server

The notification agent.

ceilometer-central

The polling agent listens to the message bus to collect data. It needs to be deployed on a Control Node. It can be deployed on the same node as *ceilometer-server*.

ceilometer-agent

The compute agents collect data from the compute nodes. They need to be deployed on all KVM compute nodes in your cloud (other hypervisors are currently not supported).

ceilometer-swift-proxy-middleware

An agent collecting data from the swift nodes. This role needs to be deployed on the same node as swift-proxy.

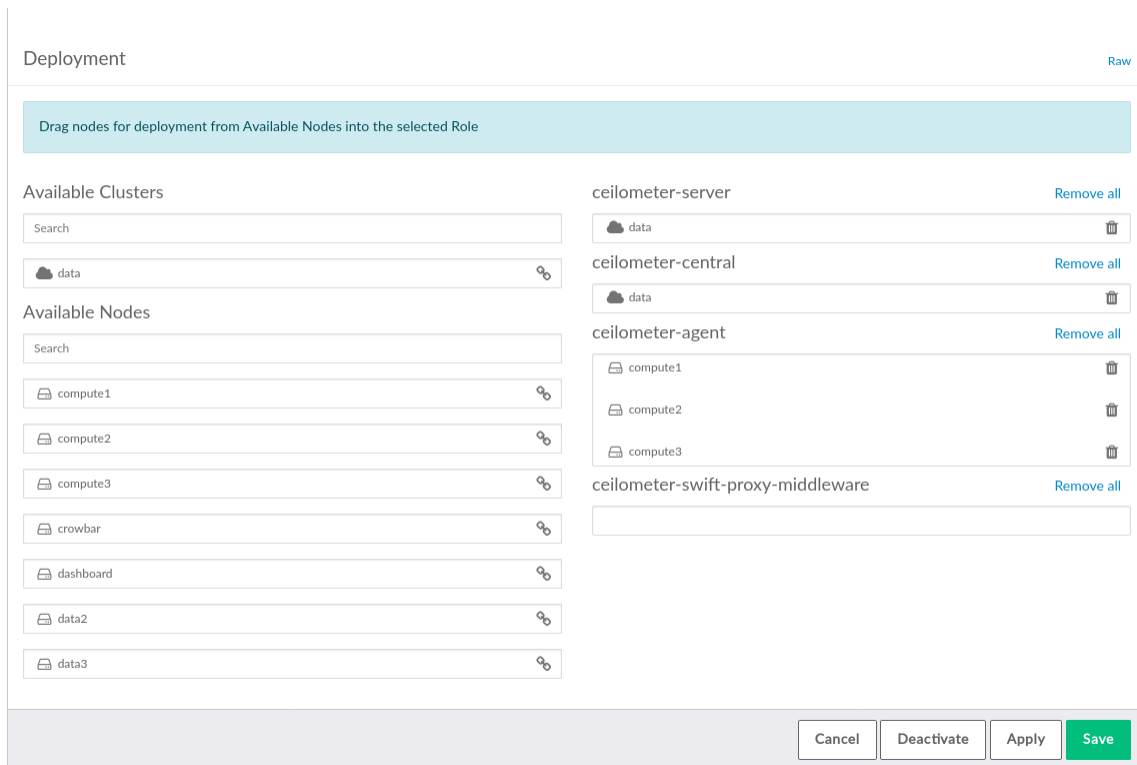


FIGURE 12.24: THE CEILOMETER BARCLAMP: NODE DEPLOYMENT

12.14.1 HA Setup for ceilometer

Making ceilometer highly available requires no special configuration—it is sufficient to deploy the roles *ceilometer-server* and *ceilometer-central* on a cluster.

12.15 Deploying manila

manila provides coordinated access to shared or distributed file systems, similar to what cinder does for block storage. These file systems can be shared between instances in SUSE OpenStack Cloud.

manila uses different back-ends. As of SUSE OpenStack Cloud Crowbar 8 currently supported back-ends include *Hitachi HNAS*, *NetApp Driver*, and *CephFS*. Two more back-end options, *Generic Driver* and *Other Driver* are available for testing purposes and are not supported.



Note: Limitations for CephFS Back-end

manila uses some CephFS features that are currently *not* supported by the SUSE Linux Enterprise Server 12 SP4 CephFS kernel client:

- RADOS namespaces
- MDS path restrictions
- Quotas

As a result, to access CephFS shares provisioned by manila, you must use `ceph-fuse`. For details, see http://docs.openstack.org/developer/manila/devref/cephfs_native_driver.html.

When first opening the manila barclamp, the default proposal *Generic Driver* is already available for configuration. To replace it, first delete it by clicking the trashcan icon and then choose a different back-end in the section *Add new manila Backend*. Select a *Type of Share* and—optionally—provide a *Name for Backend*. Activate the back-end with *Add Backend*. Note that at least one back-end must be configured.

The attributes that can be set to configure cinder depend on the back-end:

Back-end: Generic

The generic driver is included as a technology preview and is not supported.

Hitachi HNAS

Specify which EVS this backend is assigned to

Provide the name of the Enterprise Virtual Server that the selected back-end is assigned to.

Specify IP for mounting shares

IP address for mounting shares.

Specify file-system name for creating shares

Provide a file-system name for creating shares.

HNAS management interface IP

IP address of the HNAS management interface for communication between manila controller and HNAS.

HNAS username Base64 String

HNAS username Base64 String required to perform tasks like creating file-systems and network interfaces.

HNAS user password

HNAS user password. Required only if private key is not provided.

RSA/DSA private key

RSA/DSA private key necessary for connecting to HNAS. Required only if password is not provided.

The time to wait for stalled HNAS jobs before aborting

Time in seconds to wait before aborting stalled HNAS jobs.

Back-end: Netapp

Name of the Virtual Storage Server (vservers)

Host name of the Virtual Storage Server.

Server Host Name

The name or IP address for the storage controller or the cluster.

Server Port

The port to use for communication. Port 80 is usually used for HTTP, 443 for HTTPS.

User name/Password for Accessing NetApp

Login credentials.

Transport Type

Transport protocol for communicating with the storage controller or cluster. Supported protocols are HTTP and HTTPS. Choose the protocol your NetApp is licensed for.

Back-end: CephFS

Use Ceph deployed by Crowbar

Set to `true` to use Ceph deployed with Crowbar.

Back-end: Manual

Lets you manually pick and configure a driver. Only use this option for testing purposes, it is not supported.

Manila ⚠

Attributes Raw

Share backends

Backend: default

Generic backend Parameters

The generic driver can be used for testing but is not supported.

Service instance user

Service instance password
👁

Name or UUID of the service instance VM

Service network name or IP

Tenant network name or IP

Add new Manila Backend

Type of Share

Name for Backend

FIGURE 12.25: THE MANILA BARCLAMP

The manila component consists of two different roles:

manila-server

The manila server provides the scheduler and the API. Installing it on a Control Node is recommended.

manila-share

The shared storage service. It can be installed on a Control Node, but it is recommended to deploy it on one or more dedicated nodes supplied with sufficient disk space and networking capacity, since it will generate a lot of network traffic.

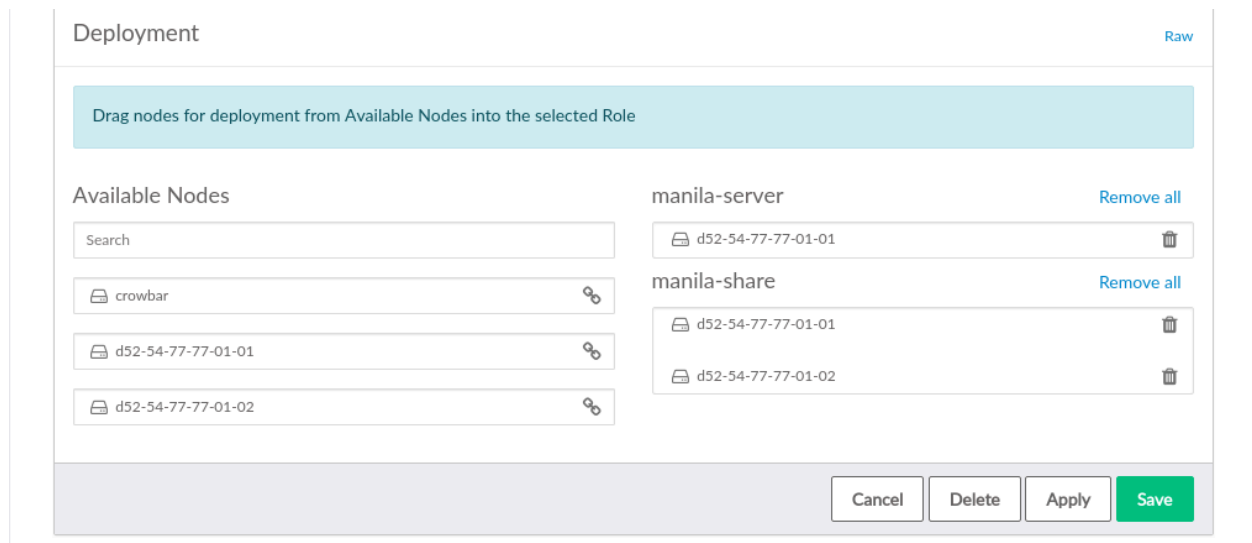


FIGURE 12.26: THE MANILA BARCLAMP: NODE DEPLOYMENT EXAMPLE

12.15.1 HA Setup for manila

While the *manila-server* role can be deployed on a cluster, deploying *manila-share* on a cluster is not supported. Therefore it is generally recommended to deploy *manila-share* on several nodes—this ensures the service continues to be available even when a node fails.

12.16 Deploying Tempest (Optional)

Tempest is an integration test suite for SUSE OpenStack Cloud written in Python. It contains multiple integration tests for validating your SUSE OpenStack Cloud deployment. For more information about Tempest refer to the OpenStack documentation at <http://docs.openstack.org/developer/tempest/>.



Important: Technology Preview

Tempest is only included as a technology preview and not supported.

Tempest may be used for testing whether the intended setup will run without problems.

It should not be used in a production environment.

Tempest should be deployed on a Control Node.

The following attributes can be configured for Tempest:

Choose User name / Password

Credentials for a regular user. If the user does not exist, it will be created.

Choose Tenant

Tenant to be used by Tempest. If it does not exist, it will be created. It is safe to stick with the default value.

Choose Tempest Admin User name/Password

Credentials for an admin user. If the user does not exist, it will be created.

Tempest !

Tempest is not supported.

Attributes Raw

Tempest user and a tenant that will be created for test run

Choose username

Choose password

Choose tenant

Deployment Raw

Drag nodes for deployment from Available Nodes into the selected Role

Available Nodes

crowbar
⌵

d52-54-77-77-01-01
⌵

d52-54-77-77-01-02
⌵

tempest Remove all

FIGURE 12.27: THE TEMPEST BARCLAMP



Tip: Running Tests

To run tests with Tempest, log in to the Control Node on which Tempest was deployed. Change into the directory `/var/lib/openstack-tempest-test`. To get an overview of available commands, run:

```
./tempest --help
```

To serially invoke a subset of all tests (“the gating smoketests”) to help validate the working functionality of your local cloud instance, run the following command. It will save the output to a log file `tempest_CURRENT_DATE.log`.

```
./tempest run --smoke --serial 2>&1 \  
| tee "tempest_$(date +%Y-%m-%d_%H%M%S).log"
```

12.16.1 HA Setup for Tempest

Tempest cannot be made highly available.

12.17 Deploying Magnum (Optional)

Magnum is an OpenStack project which offers container orchestration engines for deploying and managing containers as first class resources in OpenStack.

For more information about Magnum, see the OpenStack documentation at <http://docs.openstack.org/developer/magnum/>.

For information on how to deploy a Kubernetes cluster (either from command line or from the horizon Dashboard), see the *Supplement to Administrator Guide and User Guide*. It is available from <https://documentation.suse.com/soc/9/>.

The following *Attributes* can be configured for Magnum:

Trustee Domain: Delegate trust to cluster users if required

Deploying Kubernetes clusters in a cloud without an Internet connection requires the `registry_enabled` option in its cluster template set to `true`. To make this offline scenario work, you also need to set the *Delegate trust to cluster users if required* option to `true`. This restores the old, insecure behavior for clusters with the `registry-enabled` or `volume_driver=Rexray` options enabled.

Trustee Domain: Domain Name

Domain name to use for creating trustee for bays.

Logging: Verbose

Increases the amount of information that is written to the log files when set to `true`.

Logging: Debug

Shows debugging output in the log files when set to *true*.

Certificate Manager: Plugin

To store certificates, either use the *barbican* OpenStack service, a local directory (*Local*), or the *Magnum Database (x590keypair)*.



Note: barbican As Certificate Manager

If you choose to use barbican for managing certificates, make sure that the barbican barclamp is enabled.

Magnum !

Attributes Raw

Trustee Domain

Delegate trust to cluster users if required

false ▼

Domain Name

magnum

Certificate Manager

Plugin

Magnum Database (x509keypair) ▼

Logging

Verbose Logging

true ▼

FIGURE 12.28: THE MAGNUM BARCLAMP

The Magnum barclamp consists of the following roles: *magnum-server*. It can either be deployed on a Control Node or on a cluster—see [Section 12.17.1, “HA Setup for Magnum”](#). When deploying the role onto a Control Node, additional RAM is required for the Magnum server. It is recommended to only deploy the role to a Control Node that has 16 GB RAM.

12.17.1 HA Setup for Magnum

Making Magnum highly available requires no special configuration. It is sufficient to deploy it on a cluster.

12.18 Deploying barbican (Optional)

barbican is a component designed for storing secrets in a secure and standardized manner protected by keystone authentication. Secrets include SSL certificates and passwords used by various OpenStack components.

barbican settings can be configured in Raw mode only. To do this, open the barbican barclamp *Attribute* configuration in *Raw* mode.



FIGURE 12.29: THE BARBICAN BARCLAMP: RAW MODE

When configuring barbican, pay particular attention to the following settings:

- bind_host Bind host for the barbican API service
- bind_port Bind port for the barbican API service
- processes Number of API processes to run in Apache
- ssl Enable or disable SSL

- `threads` Number of API worker threads
- `debug` Enable or disable debug logging
- `enable_keystone_listener` Enable or disable the keystone listener services
- `kek` An encryption key (fixed-length 32-byte Base64-encoded value) for barbican's `simple_crypto` plugin. If left unspecified, the key will be generated automatically.



Note: Existing Encryption Key

If you plan to restore and use the existing barbican database after a full reinstall (including a complete wipe of the Crowbar node), make sure to save the specified encryption key beforehand. You will need to provide it after the full reinstall in order to access the data in the restored barbican database.

SSL Support: Protocol

With the default value `HTTP`, public communication will not be encrypted. Choose `HTTPS` to use SSL for encryption. See [Section 2.3, “SSL Encryption”](#) for background information and [Section 11.4.6, “Enabling SSL”](#) for installation instructions. The following additional configuration options will become available when choosing `HTTPS`:

Generate (self-signed) certificates

When set to `true`, self-signed certificates are automatically generated and copied to the correct locations. This setting is for testing purposes only and should never be used in production environments!

SSL Certificate File / SSL (Private) Key File

Location of the certificate key pair files.

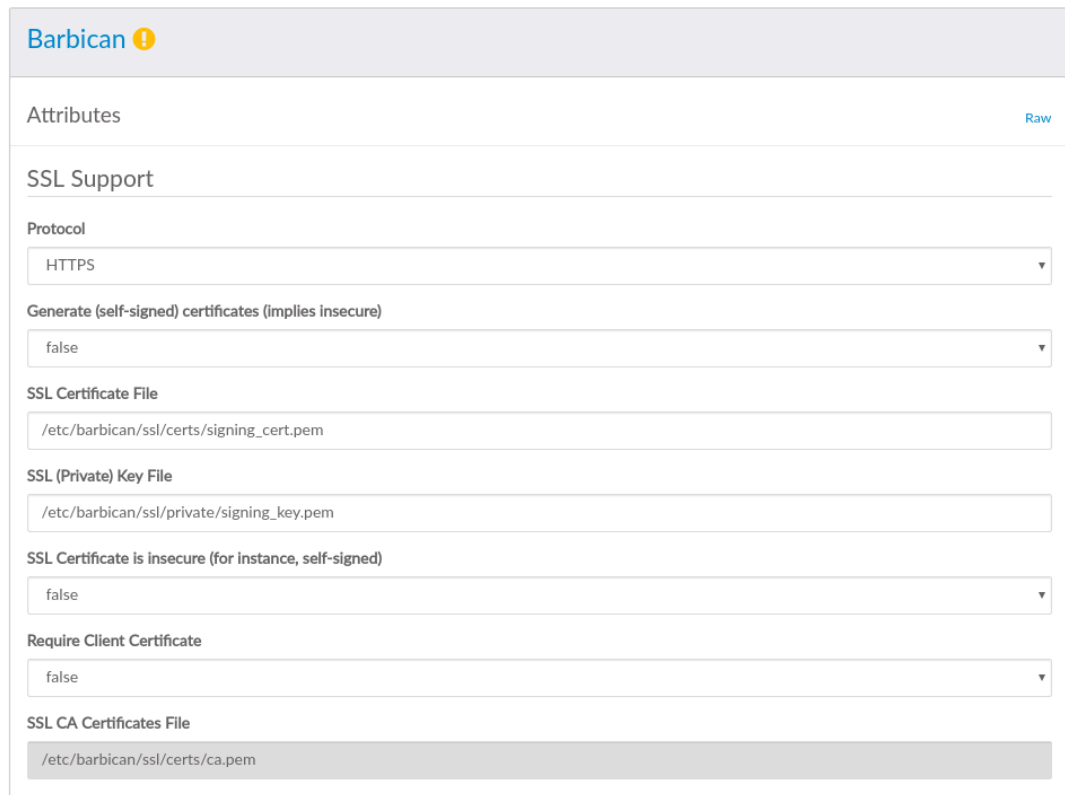
SSL Certificate is insecure

Set this option to `true` when using self-signed certificates to disable certificate checks. This setting is for testing purposes only and should never be used in production environments!

SSL CA Certificates File

Specify the absolute path to the CA certificate. This field is mandatory, and leaving it blank will cause the barclamp to fail. To fix this issue, you have to provide the absolute path to the CA certificate, restart the `apache2` service, and re-deploy the barclamp.

When the certificate is not already trusted by the pre-installed list of trusted root certificate authorities, you need to provide a certificate bundle that includes the root and all intermediate CAs.



The screenshot shows the Barbican configuration interface for SSL settings. At the top, there is a header with the Barbican logo and a warning icon. Below the header, there is a section for 'Attributes' with a 'Raw' link. The main section is titled 'SSL Support' and contains several configuration options:

- Protocol:** A dropdown menu set to 'HTTPS'.
- Generate (self-signed) certificates (implies insecure):** A dropdown menu set to 'false'.
- SSL Certificate File:** A text input field containing '/etc/barbican/ssl/certs/signing_cert.pem'.
- SSL (Private) Key File:** A text input field containing '/etc/barbican/ssl/private/signing_key.pem'.
- SSL Certificate is insecure (for instance, self-signed):** A dropdown menu set to 'false'.
- Require Client Certificate:** A dropdown menu set to 'false'.
- SSL CA Certificates File:** A text input field containing '/etc/barbican/ssl/certs/ca.pem'.

FIGURE 12.30: THE SSL DIALOG

12.18.1 HA Setup for barbican

To make barbican highly available, assign the *barbican-controller* role to the Controller Cluster.

12.19 Deploying sahara

sahara provides users with simple means to provision data processing frameworks (such as Hadoop, Spark, and Storm) on OpenStack. This is accomplished by specifying configuration parameters such as the framework version, cluster topology, node hardware details, etc.

Logging: Verbose

Set to true to increase the amount of information written to the log files.

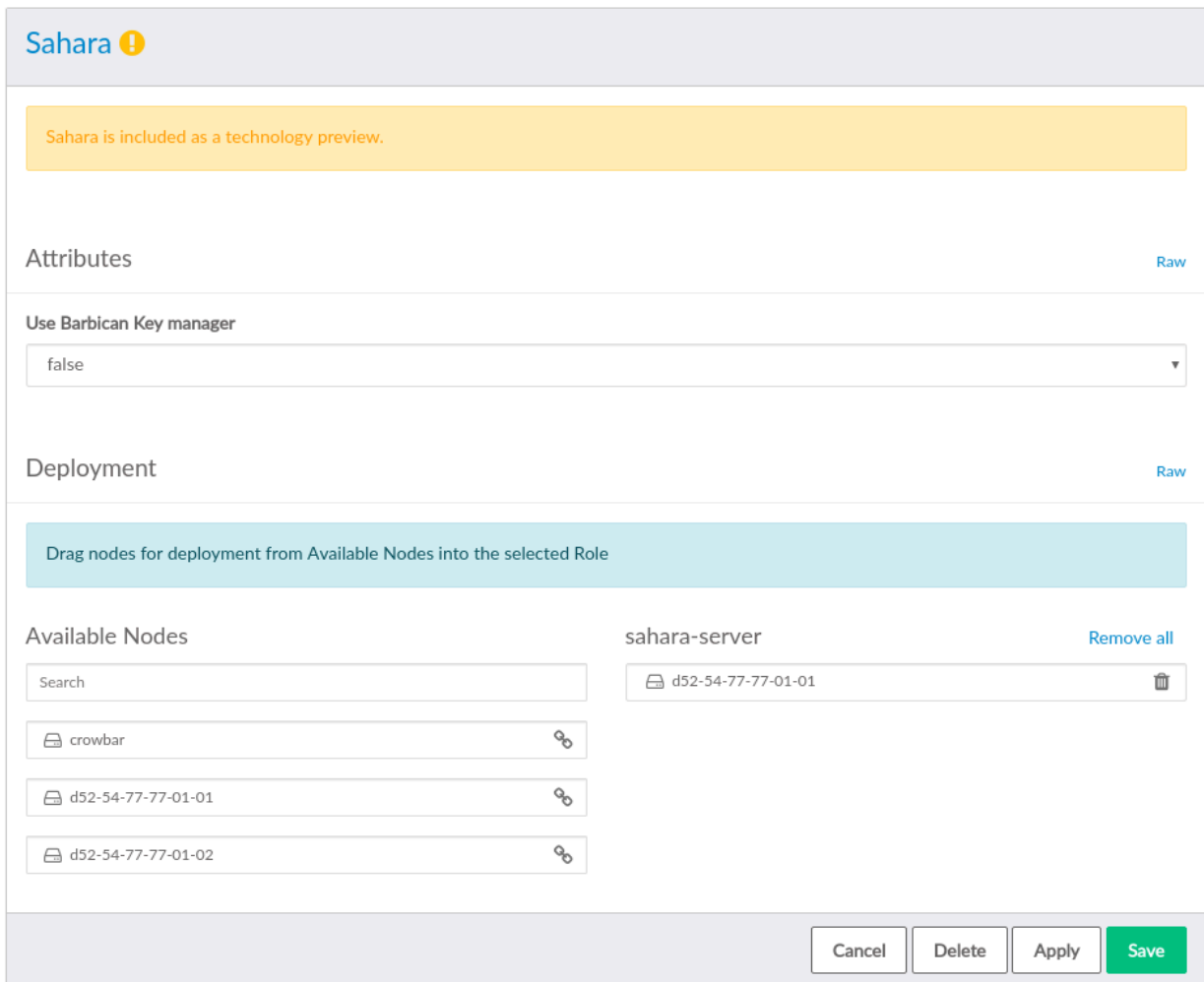


FIGURE 12.31: THE SAHARA BARCLAMP

12.19.1 HA Setup for sahara

Making sahara highly available requires no special configuration. It is sufficient to deploy it on a cluster.

12.20 Deploying Octavia

SUSE OpenStack Cloud Crowbar 9 provides Octavia Load Balancing as a Service (LBaaS). It is used to manage a fleet of virtual machines, containers, or bare metal servers—collectively known as amphorae — which it spins up on demand.



Note

Starting with the SUSE OpenStack Cloud Crowbar 9 release, we recommend running Octavia as a standalone load balancing solution. Neutron LBaaS is deprecated in the OpenStack Queens release, and Octavia is its replacement. Whenever possible, operators are strongly advised to migrate to Octavia. For further information on OpenStack Neutron LBaaS deprecation, refer to <https://wiki.openstack.org/wiki/Neutron/LBaaS/Deprecation>.



Important

Deploying the Octavia barclamp does not automatically run all tasks required to complete the migration from Neutron LBaaS.

Please refer to [Section 12.20.3, “Migrating Users to Octavia”](#) for instructions on migrating existing users to allow them to access the Octavia load balancer API after the Octavia barclamp is deployed.

Please refer to [Section 12.20.4, “Migrating Neutron LBaaS Instances to Octavia”](#) for instructions on migrating existing Neutron LBaaS load balancer instances to Octavia and on disabling the deprecated Neutron LBaaS provider after the Octavia barclamp is deployed.

Octavia consists of the following major components:

amphorae

Amphorae are the individual virtual machines, containers, or bare metal servers that accomplish the delivery of load balancing services to tenant application environments.

controller


The controller is the brains of Octavia. It consists of five sub-components as individual daemons. They can be run on separate back-end infrastructure.

- The API Controller is a subcomponent that runs Octavia’s API. It takes API requests, performs simple sanitizing on them, and ships them off to the controller worker over the Oslo messaging bus.
- The controller worker subcomponent takes sanitized API commands from the API controller and performs the actions necessary to fulfill the API request.

- The health manager subcomponent monitors individual amphorae to ensure they are up and running, and healthy. It also handles failover events if amphorae fail unexpectedly.
- The housekeeping manager subcomponent cleans up stale (deleted) database records, manages the spares pool, and manages amphora certificate rotation.
- The driver agent subcomponent receives status and statistics updates from provider drivers.

network

Octavia cannot accomplish what it does without manipulating the network environment. Amphorae are spun up with a network interface on the load balancer network. They can also plug directly into tenant networks to reach back-end pool members, depending on how any given load balancing service is deployed by the tenant.

The OpenStack Octavia team has created a glossary of terms used within the context of the Octavia project and Neutron LBaaS version 2. This glossary is available here: [Octavia Glossary \(https://docs.openstack.org/octavia/rocky/reference/glossary.html\)](https://docs.openstack.org/octavia/rocky/reference/glossary.html) .

In accomplishing its role, Octavia requires OpenStack services managed by other barclamps to be already deployed:

- Nova - For managing amphora lifecycle and spinning up compute resources on demand.
- Neutron - For network connectivity between amphorae, tenant environments, and external networks.
- Barbican - For managing TLS certificates and credentials, when TLS session termination is configured on the amphorae.
- Keystone - For authentication against the Octavia API, and for Octavia to authenticate with other OpenStack projects.
- Glance - For storing the amphora virtual machine image.

The Octavia barclamp component consists of following roles:

octavia-api

The Octavia API.

octavia-backend

Octavia worker, health-manager and house-keeping.

12.20.1 Prerequisites

Before configuring and applying the Octavia barclamp, there are a couple of prerequisites that have to be prepared: the Neutron management network used by the Octavia control plane services to communicate with Amphorae and the certificates needed to secure this communication.

12.20.1.1 Management network

Octavia needs a neutron provider network as a management network that the controller uses to communicate with the amphorae. The amphorae that Octavia deploys have interfaces and IP addresses on this network. It's important that the subnet deployed on this network be sufficiently large to allow for the maximum number of amphorae and controllers likely to be deployed throughout the lifespan of the cloud installation.

To configure the Octavia management network, the network configuration must be initialized or updated to include an `octavia` network entry. The Octavia barclamp uses this information to automatically create the neutron provider network used for management traffic.

1. If you have not yet deployed Crowbar, add the following configuration to `/etc/crowbar/network.json` to set up the Octavia management network, using the applicable VLAN ID, and network address values. If you have already deployed Crowbar, then add this configuration to the *Raw* view of the Network Barclamp.

```
"octavia": {
  "conduit": "intf1",
  "vlan": 450,
  "use_vlan": true,
  "add_bridge": false,
  "subnet": "172.31.0.0",
  "netmask": "255.255.0.0",
  "broadcast": "172.31.255.255",
  "ranges": {
    "host": { "start": "172.31.0.1",
              "end": "172.31.0.255" },
    "dhcp": { "start": "172.31.1.1",
              "end": "172.31.255.254" }
  }
},
```

Important

Care should be taken to ensure the IP subnet doesn't overlap with any of those configured for the other networks. The chosen VLAN ID must not be used within the SUSE OpenStack Cloud network and not used by neutron (i.e. if deploying neutron with VLAN support - using the plugins linuxbridge or openvswitch plus VLAN - ensure that the VLAN ID doesn't overlap with the range of VLAN IDs allocated for the `nova-fixed` neutron network).

The `host` range will be used to allocate IP addresses to the controller nodes where Octavia services are running, so it needs to accommodate the maximum number of controller nodes likely to be deployed throughout the lifespan of the cloud installation.

The `dhcp` range will be reflected in the configuration of the actual neutron provider network used for Octavia management traffic and its size will determine the maximum number of amphorae and therefore the maximum number of load balancer instances that can be running at the same time.

See [Section 7.5, "Custom Network Configuration"](#) for detailed instructions on how to customize the network configuration.

2. If Crowbar is already deployed, it is also necessary to re-apply both the neutron Barclamp and the nova Barclamp for the configuration to take effect before applying the Octavia Barclamp.

Aside from configuring the physical switches to allow VLAN traffic to be correctly forwarded, no additional external network configuration is required.

12.20.1.2 Certificates

Important

Crowbar will automatically change the filesystem ownership settings for these files to match the username and group used by the Octavia services, but it is otherwise the responsibility of the cloud administrator to ensure that access to these files on the controller nodes is properly restricted.

Octavia administrators set up certificate authorities for the two-way TLS authentication used in Octavia for command and control of amphorae. For more information, see the [Creating the Certificate Authorities](https://docs.openstack.org/octavia/stein/admin/guides/certificates.html) section of the [Octavia Certificate Configuration Guide \(https://docs.openstack.org/octavia/stein/admin/guides/certificates.html\)](https://docs.openstack.org/octavia/stein/admin/guides/certificates.html). Note that the [Configuring Octavia](#) section of that guide does not apply as the barclamp will configure Octavia.

The following certificates need to be generated and stored on all controller nodes where Octavia is deployed under `/etc/octavia/certs`, in a relative path matching the certificate location attribute values configured in the Octavia barclamp:

Server CA certificate

The Certificate Authority (CA) certificate that is used by the Octavia controller(s) to sign the generated Amphora server certificates. The Octavia control plane services also validate the server certificates presented by Amphorae during the TLS handshake against this CA certificate.

Server CA key

The private key associated with the server CA certificate. This key must be encrypted with a non-empty passphrase that also needs to be provided as a separate barclamp attribute. The private key is required alongside the server CA certificate on the Octavia controller(s), to sign the generated Amphora server certificates.

Passphrase

The passphrase used to encrypt the server CA key.

Client CA certificate

The CA certificate used to sign the client certificates installed on the Octavia controller nodes and presented by Octavia control plane services during the TLS handshake. This CA certificate is stored on the Amphorae, which use it to validate the client certificate presented by the Octavia control plane services during the TLS handshake. The same CA certificate may be used for both client and server roles, but this is perceived as a security weakness and recommended against, as a server certificate from an amphora could be used to impersonate a controller.

Client certificate concat key

The client certificate, signed with the client CA certificate, bundled together with the client certificate key, that is presented by the Octavia control plane services during the TLS handshake.

All Octavia barclamp attributes listed above, with the exception of the pasphrase are paths relative to `/etc/octavia/certs`. The required certificates must be present in their corresponding locations on all controller nodes where the Octavia barclamp will be deployed.

12.20.2 Barclamp raw mode

If a user wants to be able to debug or get access to an amphora, you can provide an SSH keyname to the barclamp via the `raw mode`. This is a keyname to a key that has been uploaded to openstack. For example:

```
openstack keypair create --public-key /etc/octavia/.ssh/id_rsa_amphora.pub  
octavia_key
```

Note that the keypair has to be owned by the octavia user.

12.20.3 Migrating Users to Octavia



Important

This behaviour is not backwards compatible with the legacy Neutron LBaaS API policy, as non-admin OpenStack users will not be allowed to run `openstack loadbalancer CLI` commands or use the load balancer horizon dashboard unless their accounts are explicitly reconfigured to be associated with one or more of these roles.



Important

Please follow the instructions documented under [Section 12.13.1, "Enabling Identity Trusts Authorization \(Optional\)"](#) on updating the trusts roles in the heat barclamp configuration. This is required to configure heat to use the correct roles when communicating with the Octavia API and manage load balancers.

Octavia employs a set of specialized roles to control access to the load balancer API:

`load-balancer_observer`

User has access to load-balancer read-only APIs.

load-balancer_global_observer

User has access to load-balancer read-only APIs including resources owned by others.

load-balancer_member

User has access to load-balancer read and write APIs.

load-balancer_quota_admin

User is considered an admin for quota APIs only.

load-balancer_admin

User is considered an admin for all load-balancer APIs including resources owned by others.

12.20.4 Migrating Neutron LBaaS Instances to Octavia



Important

Disabling LBaaS or switching the LBaaS provider in the Neutron barclamp to Octavia is not possible while there are load balancers still running under the previous Neutron LBaaS provider and will result in a Neutron barclamp redeployment failure. To avoid this, ensure that load balancer instances that are running under the old provider are either migrated or deleted.

The migration procedure documented in this section is only relevant if LBaaS was already enabled in the Neutron barclamp, with either the HAProxy or H5 provider configured, before Octavia was deployed. The procedure should be followed by operators to migrate and/or delete all load balancer instances using the Neutron LBaaS provider that are still active, and concluded the switch to Octavia by reconfiguring or disabling the deprecated Neutron LBaaS feature.

Octavia is a replacement for the Neutron LBaaS feature, that is deprecated in the SUSE OpenStack Cloud Crowbar 9 release. However, deploying the Octavia barclamp does not automatically disable the legacy Neutron LBaaS provider, if one is already configured in the Neutron barclamp. Both Octavia and Neutron LBaaS need to be enabled at the same time, to facilitate the load balancer migration process. This way, operators have a migration path they can use to gradually decommission Neutron LBaaS load balancers that use the HAProxy or F5 provider and replace them with Octavia load balancers.

With Octavia deployed and Neutron LBaaS enabled, both load balancer providers can be used simultaneously:

- The (deprecated) `neutron lbaas-...` CLI commands can be used to manage load balancer instances using the legacy Neutron LBaaS provider configured in the Neutron bar-clamp. Note that the legacy Neutron LBaaS instances will not be visible in the load balancer horizon dashboard.
- The `openstack loadbalancer` CLI commands as well as the load balancer horizon dashboard can be used to manage Octavia load balancers. Also note that OpenStack users are required to have special roles associated with their projects to be able to access the Octavia API, as covered in [Section 12.20.3, "Migrating Users to Octavia"](#).



Note

(Optional): to prevent regular users from creating or changing the configuration of currently running legacy Neutron LBaaS load balancer instances during the migration process, the neutron API policy should be temporarily changed to prevent these operations. For this purpose, a `neutron-lbaas.json` file can be created in the `/etc/neutron/policy.d` folder on all neutron-server nodes (no service restart required):

```
mkdir /etc/neutron/policy.d
cat > /etc/neutron/policy.d/neutron-lbaas.json <<EOF
{
  "context_is_admin": "role:admin",
  "context_is_advsvc": "role:advsvc",
  "default": "rule:admin_or_owner",
  "create_loadbalancer": "rule:admin_only",
  "update_loadbalancer": "rule:admin_only",
  "get_loadbalancer": "!",
  "delete_loadbalancer": "rule:admin_only",
  "create_listener": "rule:admin_only",
  "get_listener": "",
  "delete_listener": "rule:admin_only",
  "update_listener": "rule:admin_only",
  "create_pool": "rule:admin_only",
  "get_pool": "",
  "delete_pool": "rule:admin_only",
  "update_pool": "rule:admin_only",
  "create_healthmonitor": "rule:admin_only",
  "get_healthmonitor": "",
  "update_healthmonitor": "rule:admin_only",
  "delete_healthmonitor": "rule:admin_only",
}
```

```
"create_pool_member": "rule:admin_only",
"get_pool_member": "",
"update_pool_member": "rule:admin_only",
"delete_pool_member": "rule:admin_only"
}
EOF
chown -R root:neutron /etc/neutron/policy.d
chmod 640 /etc/neutron/policy.d/neutron-lbaas.json
```

If users need to create or change the configuration of currently running legacy Neutron LBaaS load balancer instances during the migration process, Create a `neutron-lbaas.json` file in the `/etc/neutron/policy.d` folder on all neutron-server nodes. The `neutron-lbaas.json` file should be empty, then restart the neutron service via **`systemctl restart openstack-neutron.service`** on all neutron-server nodes.

With all of the above in check, the actual migration process consists of replacing Neutron LBaaS instances with Octavia instances. There are many different ways to accomplish this, depending on the size and purpose of the cloud deployment, the number of load balancers that need to be migrated, the project and user configuration etc. This section only gives a few pointers and recommendations on how to approach this tasks, but the actual execution needs to be attuned to each particular situation.

Migrating a single load balancer instance is generally comprised of these steps:

- Use the `neutron lbaas-...` CLI to retrieve information about the load balancer configuration, including the complete set of related listener, pool, member and health monitor instances
- Use the `openstack loadbalancer` CLI or the load balancer horizon dashboard to create an Octavia load balancer and its associated listener, pool, member and health monitor instances to accurately match the project and Neutron LBaaS load balancer configuration extracted during the previous step. Note that the Octavia load balancer instance and the Neutron LBaaS instance cannot share the same VIP address value if both instances are running at the same time. This could be a problem, if the load balancer VIP address is accessed directly (i.e. as opposed to being accessed via a floating IP). In this case, the legacy load balancer instance needs to be deleted first, which incurs a longer interruption in service availability.

- Once the Octavia instance is up and running, if a floating IP is associated with the Neutron LBaaS load balancer VIP address, re-associate the floating IP with the Octavia load balancer VIP address. Using a floating IP has the advantage that the migration can be performed with minimal downtime. If the load balancer VIP address needs to be accessed directly (e.g. from another VM attached to the same Neutron network or router), then all the remote affected services need to be reconfigured to use the new VIP address.
- The two load balancer instances can continue to run in parallel, while the operator or owner verifies the Octavia load balancer operation. If any problems occur, the change can be reverted by undoing the actions performed during the previous step. If a floating IP is involved, this could be as simple as switching it back to the Neutron LBaaS load balancer instance.
- When it's safe, delete the Neutron LBaaS load balancer instance, along with all its related listener, pool, member and health monitor instances.

Depending on the number of load balancer instances that need to be migrated and the complexity of the overall setup that they are integrated into, the migration may be performed by the cloud operators, the owners themselves, or a combination of both. It is generally recommended that the load balancer owners have some involvement in this process or at least be notified of this migration procedure, because the load balancer migration is not an entirely seamless operation. One or more of the load balancer configuration attributes listed below may change during the migration and there may be other operational components, managed by OpenStack or otherwise (e.g. OpenStack heat stacks, configuration management scripts, database entries or non-persistent application states, etc.), that only the owner(s) may be aware of:

- The load balancer UUID value, along with the UUID values of every other related object (listeners, pools, members etc.). Even though the name values may be preserved by the migration, the UUID values will be different.
- The load balancer VIP address will change during a non-disruptive migration. This is especially relevant if there is no floating IP associated with the previous VIP address.

When the load balancer migration is complete, the Neutron LBaaS provider can either be switched to Octavia or turned off entirely in the Neutron barclamp, to finalize the migration process.

The only advantage of having Octavia configured as the Neutron LBaaS provider is that it continues to allow users to manage Octavia load balancers via the deprecated `neutron lbaas-...` CLI, but it is otherwise recommended to disable LBaaS in the Neutron barclamp.

12.21 Deploying ironic (optional)

Ironic is the OpenStack bare metal service for provisioning physical machines. Refer to the OpenStack [developer and admin manual \(https://docs.openstack.org/ironic/latest/\)](https://docs.openstack.org/ironic/latest/) for information on drivers, and administering ironic.

Deploying the ironic barclamp is done in five steps:

- Set options in the Custom view of the barclamp.
- List the `enabled_drivers` in the Raw view.
- Configure the ironic network in `network.json`.
- Apply the barclamp to a Control Node.
- Apply the `nova-compute-ironic` role to the same node you applied the ironic barclamp to, in place of the other `nova-compute-*` roles.

12.21.1 Custom View Options

Currently, there are two options in the Custom view of the barclamp.

Enable automated node cleaning

Node cleaning prepares the node to accept a new workload. When you set this to *true*, ironic collects a list of cleaning steps from the Power, Deploy, Management, and RAID interfaces of the driver assigned to the node. ironic automatically prioritizes and executes the cleaning steps, and changes the state of the node to "cleaning". When cleaning is complete the state becomes "available". After a new workload is assigned to the machine its state changes to "active".

false disables automatic cleaning, and you must configure and apply node cleaning manually. This requires the admin to create and prioritize the cleaning steps, and to set up a cleaning network. Apply manual cleaning when you have long-running or destructive tasks that you wish to monitor and control more closely. (See [Node Cleaning \(https://docs.openstack.org/ironic/latest/admin/cleaning.html\)](https://docs.openstack.org/ironic/latest/admin/cleaning.html).)

SSL Support: Protocol

SSL support is not yet enabled, so the only option is *HTTP*.

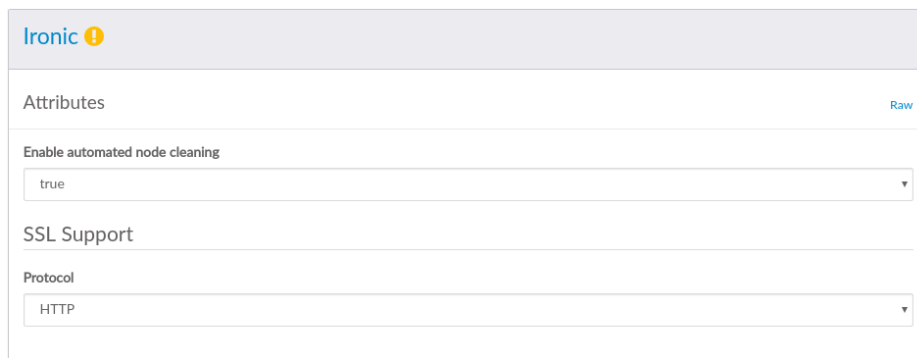


FIGURE 12.32: THE IRONIC BARCLAMP CUSTOM VIEW

12.21.2 ironic Drivers

You must enter the Raw view of barclamp and specify a list of drivers to load during service initialization. `pxe_ipmitool` is the recommended default ironic driver. It uses the Intelligent Platform Management Interface (IPMI) to control the power state of your bare metal machines, creates the appropriate PXE configurations to start them, and then performs the steps to provision and configure the machines.

```
"enabled_drivers": ["pxe_ipmitool"],
```

See [ironic Drivers \(https://docs.openstack.org/ironic/latest/admin/drivers.html\)](https://docs.openstack.org/ironic/latest/admin/drivers.html) for more information.

12.21.3 Example ironic Network Configuration

This is a complete ironic `network.json` example, using the default `network.json`, followed by a diff that shows the ironic-specific configurations.

EXAMPLE 12.1: EXAMPLE NETWORK.JSON

```
{
  "start_up_delay": 30,
  "enable_rx_offloading": true,
  "enable_tx_offloading": true,
  "mode": "single",
  "teaming": {
    "mode": 1
  },
  "interface_map": [
    {
      "bus_order": [
```



```

    "0000:00/0000:00:01",
    "0000:00/0000:00:03"
  ],
  "pattern": "PowerEdge R610"
},
{
  "bus_order": [
    "0000:00/0000:00:01.1/0000:01:00.0",
    "0000:00/0000:00:01.1/0000:01:00.1",
    "0000:00/0000:00:01.0/0000:02:00.0",
    "0000:00/0000:00:01.0/0000:02:00.1"
  ],
  "pattern": "PowerEdge R620"
},
{
  "bus_order": [
    "0000:00/0000:00:01",
    "0000:00/0000:00:03"
  ],
  "pattern": "PowerEdge R710"
},
{
  "bus_order": [
    "0000:00/0000:00:04",
    "0000:00/0000:00:02"
  ],
  "pattern": "PowerEdge C6145"
},
{
  "bus_order": [
    "0000:00/0000:00:03.0/0000:01:00.0",
    "0000:00/0000:00:03.0/0000:01:00.1",
    "0000:00/0000:00:1c.4/0000:06:00.0",
    "0000:00/0000:00:1c.4/0000:06:00.1"
  ],
  "pattern": "PowerEdge R730xd"
},
{
  "bus_order": [
    "0000:00/0000:00:1c",
    "0000:00/0000:00:07",
    "0000:00/0000:00:09",
    "0000:00/0000:00:01"
  ],
  "pattern": "PowerEdge C2100"
},
{

```

```

    "bus_order": [
      "0000:00/0000:00:01",
      "0000:00/0000:00:03",
      "0000:00/0000:00:07"
    ],
    "pattern": "C6100"
  },
  {
    "bus_order": [
      "0000:00/0000:00:01",
      "0000:00/0000:00:02"
    ],
    "pattern": "product"
  }
],
"conduit_map": [
  {
    "conduit_list": {
      "intf0": {
        "if_list": [
          "lg1",
          "lg2"
        ]
      },
      "intf1": {
        "if_list": [
          "lg1",
          "lg2"
        ]
      },
      "intf2": {
        "if_list": [
          "lg1",
          "lg2"
        ]
      },
      "intf3": {
        "if_list": [
          "lg1",
          "lg2"
        ]
      }
    },
    "pattern": "team/.*/.*"
  },
  {
    "conduit_list": {

```

```

    "intf0": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf1": {
      "if_list": [
        "?lg2"
      ]
    },
    "intf2": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf3": {
      "if_list": [
        "?lg2"
      ]
    }
  },
  "pattern": "dual/./.*"
},
{
  "conduit_list": {
    "intf0": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf1": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf2": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf3": {
      "if_list": [
        "?lg2"
      ]
    }
  },
  "pattern": "single/./.*ironic.*"
}

```

```

},
{
  "conduit_list": {
    "intf0": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf1": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf2": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf3": {
      "if_list": [
        "?lg1"
      ]
    }
  },
  "pattern": "single/././.*"
},
{
  "conduit_list": {
    "intf0": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf1": {
      "if_list": [
        "lg1"
      ]
    },
    "intf2": {
      "if_list": [
        "lg1"
      ]
    },
    "intf3": {
      "if_list": [
        "lg1"
      ]
    }
  }
}

```

```

    }
  },
  "pattern": ".*/*.*/.*"
},
{
  "conduit_list": {
    "intf0": {
      "if_list": [
        "lg1"
      ]
    },
    "intf1": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf2": {
      "if_list": [
        "?lg1"
      ]
    },
    "intf3": {
      "if_list": [
        "?lg1"
      ]
    }
  },
  "pattern": "mode/lg_adpt_count/role"
}
],
"networks": {
  "ironic": {
    "conduit": "intf3",
    "vlan": 100,
    "use_vlan": false,
    "add_bridge": false,
    "add_ovs_bridge": false,
    "bridge_name": "br-ironic",
    "subnet": "192.168.128.0",
    "netmask": "255.255.255.0",
    "broadcast": "192.168.128.255",
    "router": "192.168.128.1",
    "router_pref": 50,
    "ranges": {
      "admin": {
        "start": "192.168.128.10",
        "end": "192.168.128.11"
      }
    }
  }
}

```

```

    },
    "dhcp": {
      "start": "192.168.128.21",
      "end": "192.168.128.254"
    }
  },
  "mtu": 1500
},
"storage": {
  "conduit": "intf1",
  "vlan": 200,
  "use_vlan": true,
  "add_bridge": false,
  "mtu": 1500,
  "subnet": "192.168.125.0",
  "netmask": "255.255.255.0",
  "broadcast": "192.168.125.255",
  "ranges": {
    "host": {
      "start": "192.168.125.10",
      "end": "192.168.125.239"
    }
  }
},
"public": {
  "conduit": "intf1",
  "vlan": 300,
  "use_vlan": true,
  "add_bridge": false,
  "subnet": "192.168.122.0",
  "netmask": "255.255.255.0",
  "broadcast": "192.168.122.255",
  "router": "192.168.122.1",
  "router_pref": 5,
  "ranges": {
    "host": {
      "start": "192.168.122.2",
      "end": "192.168.122.127"
    }
  }
},
"mtu": 1500
},
"nova_fixed": {
  "conduit": "intf1",
  "vlan": 500,
  "use_vlan": true,
  "add_bridge": false,

```

```

    "add_ovs_bridge": false,
    "bridge_name": "br-fixed",
    "subnet": "192.168.123.0",
    "netmask": "255.255.255.0",
    "broadcast": "192.168.123.255",
    "router": "192.168.123.1",
    "router_pref": 20,
    "ranges": {
      "dhcp": {
        "start": "192.168.123.1",
        "end": "192.168.123.254"
      }
    },
    "mtu": 1500
  },
  "nova_floating": {
    "conduit": "intf1",
    "vlan": 300,
    "use_vlan": true,
    "add_bridge": false,
    "add_ovs_bridge": false,
    "bridge_name": "br-public",
    "subnet": "192.168.122.128",
    "netmask": "255.255.255.128",
    "broadcast": "192.168.122.255",
    "ranges": {
      "host": {
        "start": "192.168.122.129",
        "end": "192.168.122.254"
      }
    },
    "mtu": 1500
  },
  "bmc": {
    "conduit": "bmc",
    "vlan": 100,
    "use_vlan": false,
    "add_bridge": false,
    "subnet": "192.168.124.0",
    "netmask": "255.255.255.0",
    "broadcast": "192.168.124.255",
    "ranges": {
      "host": {
        "start": "192.168.124.162",
        "end": "192.168.124.240"
      }
    }
  },

```

```

    "router": "192.168.124.1"
  },
  "bmc_vlan": {
    "conduit": "intf2",
    "vlan": 100,
    "use_vlan": true,
    "add_bridge": false,
    "subnet": "192.168.124.0",
    "netmask": "255.255.255.0",
    "broadcast": "192.168.124.255",
    "ranges": {
      "host": {
        "start": "192.168.124.161",
        "end": "192.168.124.161"
      }
    }
  },
  "os_sdn": {
    "conduit": "intf1",
    "vlan": 400,
    "use_vlan": true,
    "add_bridge": false,
    "mtu": 1500,
    "subnet": "192.168.130.0",
    "netmask": "255.255.255.0",
    "broadcast": "192.168.130.255",
    "ranges": {
      "host": {
        "start": "192.168.130.10",
        "end": "192.168.130.254"
      }
    }
  },
  "admin": {
    "conduit": "intf0",
    "vlan": 100,
    "use_vlan": false,
    "add_bridge": false,
    "mtu": 1500,
    "subnet": "192.168.124.0",
    "netmask": "255.255.255.0",
    "broadcast": "192.168.124.255",
    "router": "192.168.124.1",
    "router_pref": 10,
    "ranges": {
      "admin": {
        "start": "192.168.124.10",

```



```

    },
    "pattern": "dual/./.*"
@@ -131,6 +142,36 @@
    "if_list": [
        "?lg1"
    ]
+   },
+   "intf3": {
+       "if_list": [
+           "?lg2"
+       ]
+   }
+ },
+ "pattern": "single/./.*ironic.*"
+ },
+ {
+   "conduit_list": {
+     "intf0": {
+       "if_list": [
+         "?lg1"
+       ]
+     },
+     "intf1": {
+       "if_list": [
+         "?lg1"
+       ]
+     },
+     "intf2": {
+       "if_list": [
+         "?lg1"
+       ]
+     },
+     "intf3": {
+       "if_list": [
+         "?lg1"
+       ]
+     }
+   },
+   "pattern": "single/./.*"
@@ -151,6 +192,11 @@
    "if_list": [
        "lg1"
    ]
+ },
+ "intf3": {
+   "if_list": [
+     "lg1"

```

```

+     ]
+     }
+   },
+   "pattern": ".*/*.*/.*"
@@ -171,12 +217,41 @@
+     "if_list": [
+       "?lg1"
+     ]
+   },
+   "intf3": {
+     "if_list": [
+       "?lg1"
+     ]
+   }
+ },
+ "pattern": "mode/lg_adpt_count/role"
+ }
],
"networks": {
+ "ironic": {
+   "conduit": "intf3",
+   "vlan": 100,
+   "use_vlan": false,
+   "add_bridge": false,
+   "add_ovs_bridge": false,
+   "bridge_name": "br-ironic",
+   "subnet": "192.168.128.0",
+   "netmask": "255.255.255.0",
+   "broadcast": "192.168.128.255",
+   "router": "192.168.128.1",
+   "router_pref": 50,
+   "ranges": {
+     "admin": {
+       "start": "192.168.128.10",
+       "end": "192.168.128.11"
+     },
+     "dhcp": {
+       "start": "192.168.128.21",
+       "end": "192.168.128.254"
+     }
+   },
+   "mtu": 1500
+ },
+ "storage": {
+   "conduit": "intf1",
+   "vlan": 200,

```

12.22 How to Proceed

With a successful deployment of the OpenStack Dashboard, the SUSE OpenStack Cloud Crowbar installation is finished. To be able to test your setup by starting an instance one last step remains to be done—uploading an image to the glance component. Refer to the *Supplement to Administrator Guide and User Guide*, chapter *Manage images* for instructions. Images for SUSE OpenStack Cloud can be built in SUSE Studio. Refer to the *Supplement to Administrator Guide and User Guide*, section *Building Images with SUSE Studio*.

Now you can hand over to the cloud administrator to set up users, roles, flavors, etc.—refer to the *Administrator Guide* for details. The default credentials for the OpenStack Dashboard are user name `admin` and password `crowbar`.

12.23 SUSE Enterprise Storage integration

SUSE OpenStack Cloud Crowbar supports integration with SUSE Enterprise Storage (SES), enabling Ceph block storage as well as image storage services in SUSE OpenStack Cloud.

Enabling SES Integration

To enable SES integration on Crowbar, an SES configuration file must be uploaded to Crowbar. SES integration functionality is included in the `crowbar-core` package and can be used with the Crowbar UI or CLI (`crowbarctl`). The SES configuration file describes various aspects of the Ceph environment, and keyrings for each user and pool created in the Ceph environment for SUSE OpenStack Cloud Crowbar services.

SES 7 Configuration

The following instructions detail integrating SUSE Enterprise Storage 7.0 with SUSE OpenStack Cloud.

1. Create the osd pools on the SUSE Enterprise Storage admin node (the names provided here are examples)

```
ceph osd pool create ses-cloud-volumes 16 && \  
ceph osd pool create ses-cloud-backups 16 && \  

```

```
ceph osd pool create ses-cloud-images 16 &&\
ceph osd pool create ses-cloud-vms 16
```

2. Enable the osd pools

```
ceph osd pool application enable ses-cloud-volumes rbd && \
ceph osd pool application enable ses-cloud-backups rbd && \
ceph osd pool application enable ses-cloud-images rbd && \
ceph osd pool application enable ses-cloud-vms rbd
```

3. Configure permissions on the SUSE OpenStack Cloud Crowbar admin node

```
ceph-authtool -C /etc/ceph/ceph.client.ses-cinder.keyring --name client.ses-cinder
--add-key $(ceph-authtool --gen-print-key) --cap mon "allow r" --cap osd "allow
class-read object_prefix rbd_children, allow rwx pool=ses-cloud-volumes, allow rwx
pool=ses-cloud-vms, allow rwx pool=ses-cloud-images"
ceph-authtool -C /etc/ceph/ceph.client.ses-cinder-backup.keyring --name client.ses-
cinder-backup --add-key $(ceph-authtool --gen-print-key) --cap mon "allow r" --cap
osd "allow class-read object_prefix rbd_children, allow rwx pool=ses-cloud-cinder-
backups"
ceph-authtool -C /etc/ceph/ceph.client.ses-glance.keyring --name client.ses-glance
--add-key $(ceph-authtool --gen-print-key) --cap mon "allow r" --cap osd "allow
class-read object_prefix rbd_children, allow rwx pool=ses-cloud-images"
```

4. Import the updated keyrings into Ceph

```
ceph auth import -i /etc/ceph/ceph.client.ses-cinder-backup.keyring && \
ceph auth import -i /etc/ceph/ceph.client.ses-cinder.keyring && \
ceph auth import -i /etc/ceph/ceph.client.ses-glance.keyring
```

SES 6, 5.5, 5 Configuration

For SES deployments that are version 5.5 or 6, a Salt runner is used to create all the users and pools. It also generates a YAML configuration that is needed to integrate with SUSE OpenStack Cloud. The integration runner creates separate users for cinder, cinder backup (not used by Crowbar currently) and glance. Both the cinder and nova services have the same user, because cinder needs access to create objects that nova uses.

! Important

Support for SUSE Enterprise Storage 5 and 5.5 is deprecated. The documentation for integrating these versions is included for customers who may not yet have upgraded to newer versions of SUSE Enterprise Storage . These versions are no longer officially supported.

Configure SES 6, 5.5, or 5 with the following steps:

1. Login as `root` and run the SES 5.5 Salt runner on the Salt admin host.

```
root # salt-run --out=yaml openstack.integrate prefix=mycloud
```

The prefix parameter allows pools to be created with the specified prefix. By using different prefix parameters, multiple cloud deployments can support different users and pools on the same SES deployment.

2. YAML output is created with content similar to the following example, and can be redirected to a file using the redirect operator `>` or using the additional parameter `--out-file=<filename>`:

```
ceph_conf:
  cluster_network: 10.84.56.0/21
  fsid: d5d7c7cb-5858-3218-a36f-d028df7b0673
  mon_host: 10.84.56.8, 10.84.56.9, 10.84.56.7
  mon_initial_members: ses-osd1, ses-osd2, ses-osd3
  public_network: 10.84.56.0/21
cinder:
  key: ABCDEFGaxefEMxAAW4zp2My/5HjoST2Y87654321==
  rbd_store_pool: mycloud-cinder
  rbd_store_user: cinder
cinder-backup:
  key: AQBb8hdbry2bNRAAqJC2ZzR5Q4yrionh7V5PkQ==
  rbd_store_pool: mycloud-backups
  rbd_store_user: cinder-backup
glance:
  key: AQD9eYRachg1NxAAiT6Hw/xYDA1vwSWLItLpgA==
  rbd_store_pool: mycloud-glance
  rbd_store_user: glance
nova:
  rbd_store_pool: mycloud-nova
radosgw_urls:
  - http://10.84.56.7:80/swift/v1
```

```
- http://10.84.56.8:80/swift/v1
```

3. Upload the generated YAML file to Crowbar using the UI or `crowbarctl` CLI.
4. If the Salt runner is not available, you must manually create pools and users to allow SUSE OpenStack Cloud services to use the SES/Ceph cluster. Pools and users must be created for cinder, nova, and glance. Instructions for creating and managing pools, users and keyrings can be found in the [SUSE Enterprise Storage \(https://documentation.suse.com/ses/6/\)](https://documentation.suse.com/ses/6/) Administration Guide in the Key Management section.

After the required pools and users are set up on the SUSE Enterprise Storage/Ceph cluster, create an SES configuration file in YAML format (using the example template above). Upload this file to Crowbar using the UI or `crowbarctl` CLI.

5. As indicated above, the SES configuration file can be uploaded to Crowbar using the UI or `crowbarctl` CLI.

- From the main Crowbar UI, the upload page is under *Utilities* > *SUSE Enterprise Storage*.

If a configuration is already stored in Crowbar, it will be visible in the upload page. A newly uploaded configuration will replace existing one. The new configuration will be applied to the cloud on the next `chef-client` run. There is no need to reapply proposals.

Configurations can also be deleted from Crowbar. After deleting a configuration, you must manually update and reapply all proposals that used SES integration.

- With the `crowbarctl` CLI, the command `crowbarctl ses upload FILE` accepts a path to the SES configuration file.

Cloud Service Configuration

SES integration with SUSE OpenStack Cloud services is implemented with relevant Barclamps and installed with the `crowbar-openstack` package.

glance

Set Use SES Configuration to `true` under RADOS Store Parameters. The glance barclamp pulls the uploaded SES configuration from Crowbar when applying the glance proposal and on `chef-client` runs. If the SES configuration is uploaded before the glance proposal is created, Use SES Configuration is enabled automatically upon proposal creation.

cinder

Create a new RADOS backend and set `Use SES Configuration` to `true`. The cinder barclamp pulls the uploaded SES configuration from Crowbar when applying the cinder proposal and on `chef-client` runs. If the SES configuration was uploaded before the cinder proposal was created, a `ses-ceph` RADOS backend is created automatically on proposal creation with `Use SES Configuration` already enabled.

nova

To connect with volumes stores in SES, nova uses the configuration from the cinder barclamp. For ephemeral storage, nova re-uses the `rbd_store_user` and `key` from cinder but has a separate `rbd_store_pool` defined in the SES configuration. Ephemeral storage on SES can be enabled or disabled by setting `Use Ceph RBD Ephemeral Backend` in nova proposal. In new deployments it is enabled by default. In existing ones it is disabled for compatibility reasons.

RADOS Gateway Integration

Besides block storage, the SES cluster can also be used as a swift replacement for object storage. If `radosgw_urls` section is present in uploaded SES configuration, first of the URLs is registered in the keystone catalog as the "swift"/"object-store" service. Some configuration is needed on SES side to fully integrate with keystone auth. If SES integration is enabled on a cloud with swift deployed, SES object storage service will get higher priority by default. To override this and use swift for object storage instead, remove `radosgw_urls` section from the SES configuration file and re-upload it to Crowbar. Re-apply swift proposal or wait for next periodic chef-client run to make changes effective.

12.24 Roles and Services in SUSE OpenStack Cloud Crowbar

The following table lists all roles (as defined in the barclamps), and their associated services. As of SUSE OpenStack Cloud Crowbar 8, this list is work in progress. Services can be manually started and stopped with the commands `systemctl start SERVICE` and `systemctl stop SERVICE`.

Role	Service
ceilometer-agent	<u>openstack-ceilometer-agent-compute</u>
ceilometer-central	<u>openstack-ceilometer-agent-notification</u>
ceilometer-server	<u>openstack-ceilometer-agent-central</u>
ceilometer-swift-proxy-middle-ware	
cinder-controller	<u>openstack-cinder-api</u>
	<u>openstack-cinder-scheduler</u>
cinder-volume	<u>openstack-cinder-volume</u>
database-server	<u>postgresql</u>
glance-server	<u>openstack-glance-api</u>
	<u>openstack-glance-registry</u>
heat-server	<u>openstack-heat-api-cfn</u>
	<u>openstack-heat-api-cloudwatch</u>
	<u>openstack-heat-api</u>
	<u>openstack-heat-engine</u>
horizon	<u>apache2</u>
keystone-server	<u>openstack-keystone</u>
manila-server	<u>openstack-manila-api</u>
	<u>openstack-manila-scheduler</u>
manila-share	<u>openstack-manila-share</u>
neutron-server	<u>openstack-neutron</u>

Role	Service
nova-compute-*	openstack-nova-compute
	openstack-neutron-openvswitch-agent (when neutron is deployed with openvswitch)
nova-controller	openstack-nova-api
	openstack-nova-cert
	openstack-nova-conductor
	openstack-nova-novncproxy
	openstack-nova-objectstore
	openstack-nova-scheduler
rabbitmq-server	rabbitmq-server
swift-dispersion	none
swift-proxy	openstack-swift-proxy
swift-ring-compute	none
swift-storage	openstack-swift-account-auditor
	openstack-swift-account-reaper
	openstack-swift-account-replicator
	openstack-swift-account
	openstack-swift-container-auditor
	openstack-swift-container-replicator
	openstack-swift-container-sync
	openstack-swift-container-updater

Role	Service
	<u>openstack-swift-container</u>
	<u>openstack-swift-object-auditor</u>
	<u>openstack-swift-object-expirer</u>
	<u>openstack-swift-object-replicator</u>
	<u>openstack-swift-object-updater</u>
	<u>openstack-swift-object</u>

12.25 Crowbar Batch Command

This is the documentation for the **crowbar batch** subcommand.

crowbar batch provides a quick way of creating, updating, and applying Crowbar proposals. It can be used to:

- Accurately capture the configuration of an existing Crowbar environment.
- Drive Crowbar to build a complete new environment from scratch.
- Capture one SUSE OpenStack Cloud environment and then reproduce it on another set of hardware (provided hardware and network configuration match to an appropriate extent).
- Automatically update existing proposals.

As the name suggests, **crowbar batch** is intended to be run in “batch mode” that is mostly unattended. It has two modes of operation:

crowbar batch export

Exports a YAML file which describes existing proposals and how their parameters deviate from the default proposal values for that barclamp.

crowbar batch build

Imports a YAML file in the same format as above. Uses it to build new proposals if they do not yet exist. Updates the existing proposals so that their parameters match those given in the YAML file.

12.25.1 YAML file format

Here is an example YAML file. At the top-level there is a proposals array, each entry of which is a hash representing a proposal:

```
proposals:
- barclamp: provisioner
  # Proposal name defaults to 'default'.
  attributes:
    shell_prompt: USER@ALIAS:CWD SUFFIX
- barclamp: database
  # Default attributes are good enough, so we just need to assign
  # nodes to roles:
  deployment:
    elements:
      database-server:
        - "@@controller1@"
- barclamp: rabbitmq
  deployment:
    elements:
      rabbitmq-server:
        - "@@controller1@"
```



Note: Reserved Indicators in YAML

Note that the characters `@` and ``` are reserved indicators in YAML. They can appear anywhere in a string *except at the beginning*. Therefore a string such as `@@controller1@` needs to be quoted using double quotes.

12.25.2 Top-level proposal attributes

barclamp

Name of the barclamp for this proposal (required).

name

Name of this proposal (optional; default is `default`). In **build** mode, if the proposal does not already exist, it will be created.

attributes

An optional nested hash containing any attributes for this proposal which deviate from the defaults for the barclamp.

In **export** mode, any attributes set to the default values are excluded to keep the YAML as short and readable as possible.

In **build** mode, these attributes are deep-merged with the current values for the proposal. If the proposal did not already exist, batch build will create it first. The attributes are merged with the default values for the barclamp's proposal.

wipe_attributes

An optional array of paths to nested attributes which should be removed from the proposal. Each path is a period-delimited sequence of attributes; for example `pacemaker.stonith.sbd.nodes` would remove all SBD nodes from the proposal if it already exists. If a path segment contains a period, it should be escaped with a backslash, for example `segment-one.segment\ .two.segment_three`.

This removal occurs before the deep merge described above. For example, think of a YAML file which includes a Pacemaker barclamp proposal where the `wipe_attributes` entry contains `pacemaker.stonith.sbd.nodes`. A batch build with this YAML file ensures that only SBD nodes listed in the `attributes_sibling` hash are used at the end of the run. In contrast, without the `wipe_attributes` entry, the given SBD nodes would be appended to any SBD nodes already defined in the proposal.

deployment

A nested hash defining how and where this proposal should be deployed.

In **build** mode, this hash is deep-merged in the same way as the attributes hash, except that the array of elements for each Chef role is reset to the empty list before the deep merge. This behavior may change in the future.

12.25.3 Node Alias Substitutions

A string like `@@node@@` (where `node` is a node alias) will be substituted for the name of that node, no matter where the string appears in the YAML file. For example, if `controller1` is a Crowbar alias for node `d52-54-02-77-77-02.mycloud.com`, then `@@controller1@@` will be substituted for that host name. This allows YAML files to be reused across environments.

12.25.4 Options

In addition to the standard options available to every `crowbar` subcommand (run `crowbar batch --help` for a full list), there are some extra options specifically for `crowbar batch`:

`--include <barclamp[.proposal]>`

Only include the barclamp / proposals given.

This option can be repeated multiple times. The inclusion value can either be the name of a barclamp (for example, `pacemaker`) or a specifically named proposal within the barclamp (for example, `pacemaker.network_cluster`).

If it is specified, then only the barclamp / proposals specified are included in the build or export operation, and all others are ignored.

`--exclude <barclamp[.proposal]>`

This option can be repeated multiple times. The exclusion value is the same format as for `--include`. The barclamps / proposals specified are excluded from the build or export operation.

`--timeout <seconds>`

Change the timeout for Crowbar API calls.

As Chef's run lists grow, some of the later OpenStack barclamp proposals (for example `nova`, `horizon`, or `heat`) can take over 5 or even 10 minutes to apply. Therefore you may need to increase this timeout to 900 seconds in some circumstances.

13 Limiting Users' Access Rights

To limit users' access rights (or to define more fine-grained access rights), you can use Role Based Access Control (RBAC, only available with keystone v3). In the example below, we will create a new role (`ProjectAdmin`). It allows users with this role to add and remove other users to the `member` role on the same project.

To create a new role that can be assigned to a user-project pair, the following basic steps are needed:

1. Create a custom `policy.json` file for the keystone component. On the node where the `keystone-server` role is deployed, copy the file to `/etc/keystone/CUSTOM_policy.json`. For details, see [Section 13.1, "Editing policy.json"](#).
2. Create a custom `keystone_policy.json` file for the horizon component. On the node where the `nova_dashboard-server` role is deployed, copy the custom `keystone_policy.json` file to `/srv/www/openstack-dashboard/openstack_dashboard/conf/` (default directory for policy files in horizon). For details, see [Section 13.2, "Editing keystone_policy.json"](#).
3. Make the keystone component aware of the `CUSTOM_policy.json` file by editing and reapplying the `keystone` barclamp. For details, see [Section 13.3, "Adjusting the keystone Barclamp Proposal"](#).
4. Make the horizon component aware of the `keystone_policy.json` file by editing and reapplying the `horizon` barclamp. For details, see [Section 13.4, "Adjusting the horizon Barclamp Proposal"](#).

13.1 Editing policy.json

The `policy.json` file is located in `/etc/keystone/` on the node where the `keystone-server` role is deployed.

1. Copy `/etc/keystone/policy.json` and save it under a different name, for example `CUSTOM_policy.json`.

! Important: Use Different File Name

If you use the same name as the original file, your custom file will be overwritten by the next package update.

2. To add the new role, enter the following two lines at the beginning of the file:

```
{
  "subadmin": "role:ProjectAdmin",
  "projectadmin": "rule:subadmin and project_id:%(target.project.id)s",
  [...]
}
```

3. Adjust the other rules in the file accordingly:

```
"identity:get_domain": "rule:admin_required or rule:subadmin",
[...]
"identity:get_project": "rule:admin_required or rule:projectadmin",
[...]
"identity:list_user_projects": "rule:admin_or_owner or rule:projectadmin",
[...]
"identity:update_project": "rule:admin_required or rule:projectadmin",
[...]
"identity:get_user": "rule:admin_required or rule:projectadmin",
"identity:list_users": "rule:admin_required or rule:subadmin",
[...]
"identity:list_groups": "rule:admin_required or rule:subadmin",
[...]
"identity:list_roles": "rule:admin_required or rule:subadmin",
[...]
"identity:list_grants": "rule:admin_required or (rule:subadmin and project_id:
%(target.project.id)s)",
  "identity:create_grant": "rule:admin_required or (rule:subadmin and project_id:
%(target.project.id)s and 'member':%(target.role.name)s)",
  "identity:revoke_grant": "rule:admin_required or (rule:subadmin and project_id:
%(target.project.id)s and 'member':%(target.role.name)s)",
[...]
"identity:list_role_assignments": "rule:admin_required or rule:subadmin",
```

4. Save the changes.
5. On the node where the `keystone-server` role is deployed, copy the file to `/etc/keystone/CUSTOM_policy.json`. Usually, the `keystone-server` role is deployed to a Control Node (or to a cluster, if you use a High Availability setup).

13.2 Editing keystone_policy.json

By default, the `keystone_policy.json` file is located in `/srv/www/openstack-dashboard/openstack_dashboard/conf/` on the node where the `nova_dashboard-server` role is deployed. It is similar (but not identical) to `policy.json` and defines which actions the user with a certain role is allowed to execute in horizon. If the user is not allowed to execute a certain action, the OpenStack Dashboard will show an error message.

1. Copy `/srv/www/openstack-dashboard/openstack_dashboard/conf/keystone_policy.json` and save it under a different name, for example `CUSTOM_keystone_policy.json`.



Important: Use Different File Name

If you use the same name as the original file, your custom file will be overwritten by the next package update.

2. To add the new role, enter the following two lines at the beginning of the file:

```
{
  "subadmin": "role:ProjectAdmin",
  "projectadmin": "rule:subadmin and project_id:$(target.project.id)s",
  [...]
```

3. Adjust the other rules in the file accordingly:

```
"identity:get_project": "rule:admin_required or rule:projectadmin",
[...]
"identity:list_user_projects": "rule:admin_or_owner or rule:projectadmin",
[...]
"identity:get_user": "rule:admin_required or rule:projectadmin",
"identity:list_users": "rule:admin_required or rule:subadmin",
[...]
"identity:list_roles": "rule:admin_required or rule:subadmin",
[...]
"identity:list_role_assignments": "rule:admin_required or rule:subadmin",
```

4. Save the changes and copy the file to `/srv/www/openstack-dashboard/openstack_dashboard/conf/CUSTOM_keystone_policy.json` on the node where the `nova_dashboard-server` role is deployed.

13.3 Adjusting the *keystone* Barclamp Proposal

1. Log in to the Crowbar Web interface.
2. Select *Barclamps* > *All barclamps*.
3. Go to the *keystone* barclamp and click *Edit*.
4. In the *Attributes* section, click *Raw*. This shows the complete configuration file and allows you to edit it directly.
5. Adjust the `policy_file` parameter to point to the `CUSTOM_policy.json` file. For example:

```
{
  [...]
  "policy_file": "mypolicy.json",
```

6. *Save* and *Apply* the changes to the *keystone* barclamp.

13.4 Adjusting the *horizon* Barclamp Proposal

1. Log in to the Crowbar Web interface.
2. Select *Barclamps* > *All barclamps*.
3. Go to the *horizon* barclamp and click *Edit*.
4. In the *Attributes* section, click *Raw*. This shows the complete configuration file and allows you to edit it directly.
5. If needed, adjust the `policy_file_path` parameter to point to the directory where you copied the newly added `CUSTOM_keystone_policy.json` file. By default, its value is an empty string—this means that the default directory will be used.
6. Enter the new file's name as value of the `identity` parameter within the `policy_file` section (❶):

```
{
  "policy_file_path": "",
  "policy_file": {
    "identity": "mykeystone_policy.json", ❶
    "compute": "nova_policy.json",
    "volume": "cinder_policy.json",
```

```
"image": "glance_policy.json",
"orchestration": "heat_policy.json",
"network": "neutron_policy.json",
"telemetry": "ceilometer_policy.json"
```

7. *Save* and *Apply* the changes to the horizon barclamp.

13.5 Pre-Installed Service Admin Role Components

The following are the roles defined in SUSE OpenStack Cloud Crowbar. These roles serve as a way to group common administrative needs at the OpenStack service level. Each role represents administrative privilege into each service. Multiple roles can be assigned to a user. You can assign a Service Admin Role to a user once you have determined that the user is authorized to perform administrative actions and access resources in that service.

The main components of Service Administrator Roles are:

- `nova_admin` role in the identity service (keystone) and support in `nova_policy.json`. Assign this role to users whose job function it is to perform `nova-compute`-related administrative tasks.
- `neutron_admin` role in the identity service and support in `neutron_policy.json`. Assign this role to users whose job function it is to perform neutron networking-related administrative tasks.
- `cinder_admin` role in the identity service and support in `cinder_policy.json`. Assign this role to users whose job function it is to perform cinder storage-related administrative tasks.
- `glance_admin` role in the identity service and support in `glance_policy.json`. Assign this role to users whose job function it is to perform cinder storage-related administrative tasks.



Warning: Changing `glance_policy.json` may Introduce a Security Issue

The OpenStack Security Note OSSN-0075 <https://wiki.openstack.org/wiki/OSSN/OSSN-0075> describes a scenario where a malicious tenant is able to reuse deleted glance image IDs to share malicious images with other tenants in a manner that is undetectable to the victim tenant.

The default policy `glance_policy.json` that is shipped with SUSE OpenStack Cloud Crowbar prevents this by ensuring only admins can deactivate/reactivate images:

```
"deactivate": "role:admin"  
"reactivate": "role:admin"
```

SUSE suggests these settings should *not* be changed. If you do change them please refer to the OSSF-0075 <https://wiki.openstack.org/wiki/OSSF/OSSF-0075> for details on the exact scope of the security issue.

14 Configuration Files for OpenStack Services

Typically, each OpenStack component comes with a configuration file, for example: `/etc/nova/nova.conf`.

These configuration files can still be used. However, to configure an OpenStack component and its different components and roles, it is now preferred to add custom configuration file snippets to a `SERVICE.conf.d/` directory instead.

14.1 Default Configuration Files

By default, a configuration snippet with a basic configuration for each OpenStack component is available in the following directory:

```
/etc/SERVICE/SERVICE.conf.d/010-SERVICE.conf
```

For example: `/etc/nova/nova.conf.d/010-nova.conf`

Those files should not be modified.

14.2 Custom Configuration Files

To adjust or overwrite settings for the respective OpenStack component, add a custom configuration file to the same directory, `/etc/SERVICE/SERVICE.conf.d/`.

The same applies if you want to configure individual components or roles of an OpenStack component, such as `nova-api` or `nova-compute`, for example. But in this case, add your custom configuration file to the following directory:

```
/etc/SERVICE/ROLE.conf.d/
```

For example: `/etc/nova/nova-api.conf.d/`

All custom configuration file must follow the rules listed in [Section 14.3, “Naming Conventions for Custom Configuration Files”](#).

14.3 Naming Conventions for Custom Configuration Files

Use the following rules for any configuration files you add:

- The file name must start with a 3-digit number and a dash. For example: `/etc/nova/nova.conf.d/500-nova.conf`
- The file must have the following file name extension: `.conf`
- For configuration management systems (for example: Crowbar, Salt), use numbers between `100` and `499`.
- To override settings written by the configuration management system, use numbers starting from `500`. They have higher priority.

14.4 Processing Order of Configuration Files

The configuration files are processed in the following order:

- `/etc/SERVICE/SERVICE.conf`
- `/etc/SERVICE/SERVICE.conf.d/*.conf` (in dictionary order)
- `/etc/SERVICE/ROLE.conf.d/*.conf` (in dictionary order)

If conflicting values are set for the same parameter, the last configured value overwrites all previous ones. In particular, values defined in

```
/etc/SERVICE/SERVICE.conf.d/XXX-SERVICE.conf
```

overwrite configuration values in

```
/etc/SERVICE/SERVICE.conf
```

14.5 Restarting with New or Changed Configuration Files

After making changes to configuration files, you must restart the service(s) where the configuration changes should apply.

For example: to restart the nova service(s) with a new configuration, run the following command:

```
systemctl restart openstack-nova-compute
```

14.6 For More Information

For details, also see [/etc/SERVICE/README.config](#).

15 Installing SUSE CaaS Platform heat Templates

This chapter describes how to install SUSE CaaS Platform heat template on SUSE OpenStack Cloud Crowbar.

15.1 SUSE CaaS Platform heat Installation Procedure

PROCEDURE 15.1: PREPARATION

1. Download the latest SUSE CaaS Platform for OpenStack image (for example, `SUSE-CaaS-Platform-3.0-OpenStack-Cloud.x86_64-1.0.0-GM.qcow2`) from <https://download.suse.com>.

2. Upload the image to glance:

```
openstack image create --public --disk-format qcow2 --container-format \
bare --file SUSE-CaaS-Platform-3.0-OpenStack-Cloud.x86_64-1.0.0-GM.qcow2 \
CaaS-3
```

3. Install the `caasp-openstack-heat-templates` package on a machine with SUSE OpenStack Cloud Crowbar repositories:

```
zypper in caasp-openstack-heat-templates
```

The installed templates are located in `/usr/share/caasp-openstack-heat-templates`. Alternatively, you can get official heat templates by cloning the appropriate Git repository:

```
git clone https://github.com/SUSE/caasp-openstack-heat-templates
```

PROCEDURE 15.2: INSTALLING TEMPLATES VIA HORIZON

1. In horizon, go to *Project > Stacks > Launch Stack*.
2. Select *File* from the *Template Source* drop-down box and upload the `caasp-stack.yaml` file.
3. In the *Launch Stack* dialog, provide the required information (stack name, password, flavor size, external network of your environment, etc.).
4. Click *Launch* to launch the stack. This creates all required resources for running SUSE CaaS Platform in an OpenStack environment. The stack creates one Admin Node, one Master Node, and server worker nodes as specified.

PROCEDURE 15.3: INSTALL TEMPLATES FROM THE COMMAND LINE

1. Specify the appropriate flavor and network settings in the `caasp-environment.yaml` file.
2. Create a stack in heat by passing the template, environment file, and parameters:

```
openstack stack create -t caasp-stack.yaml -e caasp-environment.yaml \
--parameter image=CaaS-3 caasp-stack
```

PROCEDURE 15.4: ACCESSING VELUM SUSE CAAS PLATFORM DASHBOARD

1. After the stack has been created, the Velum SUSE CaaS Platform dashboard runs on the Admin Node. You can access it using the Admin Node's floating IP address.
2. Create an account and follow the steps in the Velum SUSE CaaS Platform dashboard to complete the SUSE CaaS Platform installation.

When you have successfully accessed the admin node web interface via the floating IP, follow the instructions at <https://documentation.suse.com/suse-caasp/3/single-html/caasp-deployment/> to continue the setup of SUSE CaaS Platform.

15.2 Installing SUSE CaaS Platform with Multiple Masters



Note

A heat stack with load balancing and multiple master nodes can only be created from the command line, because horizon does not have support for nested heat templates.

Install the `caasp-openstack-heat-templates` package on a machine with SUSE OpenStack Cloud Crowbar repositories:

```
zypper in caasp-openstack-heat-templates
```

The installed templates are located in `/usr/share/caasp-openstack-heat-templates`.

A working load balancer is needed in your SUSE OpenStack Cloud deployment. SUSE OpenStack Cloud Crowbar uses HAProxy.

Verify that load balancing with HAProxy is working correctly in your OpenStack installation by creating a load balancer manually and checking that the `provisioning_status` changes to `Active`.

```
tux > openstack loadbalancer show  
<LOAD_BALANCER_ID>
```

HAProxy is the default load balancer provider in SUSE OpenStack Cloud Crowbar.

The steps below can be used to set up a network, subnet, router, security and IPs for a test `lb_net1` network with `lb_subnet1` subnet.

```
tux > openstack network create lb_net1  
  tux > openstack subnet create --name lb_subnet1 lb_net1 \  
--subnet-range 172.29.0.0/24 --gateway 172.29.0.2  
tux > openstack router create lb_router1  
tux > openstack router add subnet lb_router1 lb_subnet1  
tux > openstack router set lb_router1 --external-gateway ext-net  
tux > openstack network list
```

PROCEDURE 15.5: STEPS TO INSTALL SUSE CAAS PLATFORM WITH MULTIPLE MASTERS

1. Specify the appropriate flavor and network settings in the `caasp-multi-master-environment.yaml` file.
2. Set `master_count` to the desired number in the `caasp-multi-master-environment.yaml` file. The master count must be set to an odd number of nodes.

```
master_count: 3
```

3. Create a stack in heat by passing the template, environment file, and parameters:

```
tux > openstack stack create -t caasp-multi-master-stack.yaml \  
-e caasp-multi-master-environment.yaml --parameter image=CaaSP-3 caasp-multi-master-  
stack
```

4. Find the floating IP address of the load balancer. This is necessary for accessing the Velum SUSE CaaS Platform dashboard.

- a. `tux > openstack loadbalancer list --provider`

- b. From the output, copy the id and enter it in the following command as shown in the following example:

```
tux > openstack loadbalancer show id
```

```
+-----+-----+
| Field          | Value                                |
+-----+-----+
| admin_state_up | True                                 |
| description    |                                     |
| id             | 0d973d80-1c79-40a4-881b-42d111ee9625 |
| listeners      | {"id": "c9a34b63-a1c8-4a57-be22-75264769132d"} |
|                | {"id": "4fa2dae0-126b-4eb0-899f-b2b6f5aab461"} |
| name           | caasp-stack-master_lb-bhr66gtrx3ue  |
| operating_status | ONLINE                              |
| pools          | {"id": "8c011309-150c-4252-bb04-6550920e0059"} |
|                | {"id": "c5f55af7-0a25-4dfa-a088-79e548041929"} |
| provider       | haproxy                             |
| provisioning_status | ACTIVE                              |
| tenant_id      | fd7ffc07400642b1b05dbef647deb4c1   |
| vip_address     | 172.28.0.6                          |
| vip_port_id    | 53ad27ba-1ae0-4cd7-b798-c96b53373e8b |
| vip_subnet_id  | 87d18a53-ad0c-4d71-b82a-050c229b710a |
+-----+-----+
```

- c. Search the floating IP list for vip_address

```
tux > openstack floating ip list | grep 172.28.0.6
| d636f3...481b0c | fd7ff...deb4c1 | 172.28.0.6 | 10.84.65.37 |
| 53ad2...373e8b |
```

- d. The load balancer floating ip address is 10.84.65.37

Accessing the Velum SUSE CaaS Platform Dashboard

After the stack has been created, the Velum SUSE CaaS Platform dashboard runs on the admin node. You can access it using the floating IP address of the admin node.

Create an account and follow the steps in the Velum SUSE CaaS Platform dashboard to complete the SUSE CaaS Platform installation.

SUSE CaaS Platform Admin Node Install: Screen 1

If you plan to manage your containers using Helm or Airship (this is common), check the box labeled Install Tiller (Helm's server component).

Welcome! You have signed up successfully. ✕

Initial CaaS Platform Configuration

Generic settings

Internal Dashboard Location
 ?

Cluster services

Install Tiller (Helm's server component)

Overlay network settings Show

Proxy settings Enable Disable

SUSE registry mirror Enable Disable

Cloud provider integration ? Enable Disable

Container runtime ?

The choice of container runtime is completely transparent to end-users of the cluster. Neither Kubernetes manifests nor container images need to be changed.

Choose the runtime

Docker open source engine cri-o

Docker open source engine (default) is a production-ready runtime, fully supported by SUSE.

System wide certificate Show

Next

SUSE CaaS Platform Admin Node Install: Screen 2

Bootstrap your CaaS Platform

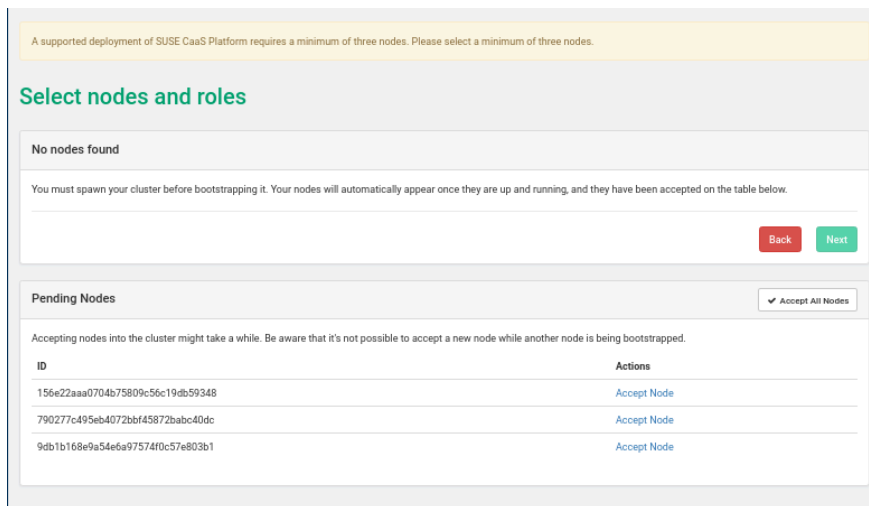
In order to complete the installation, it is necessary to bootstrap a few additional nodes, those will be the Kubernetes Master and Workers. This process leverages AutoYaST and is (almost) fully automated. In case you are not familiar with it, you can find more information about AutoYaST in the [official documentation](#). The automatic installation gets invoked by adding `autoyast=http://10.84.65.17/autoyast` to the kernel parameter list. If you aren't under a PXE environment you can also use `netsetup=dhcp` kernel parameter for the network to be automatically configured using a reachable DHCP server. As installation media, you can use the very same image you bootstrapped the admin node with. A ready to use AutoYaST profile has already been generated for you during the bootstrap of the admin node. Bootstrap all the nodes you want to make part of this platform by adding the following boot parameter `autoyast=http://10.84.65.17/autoyast`

Tips

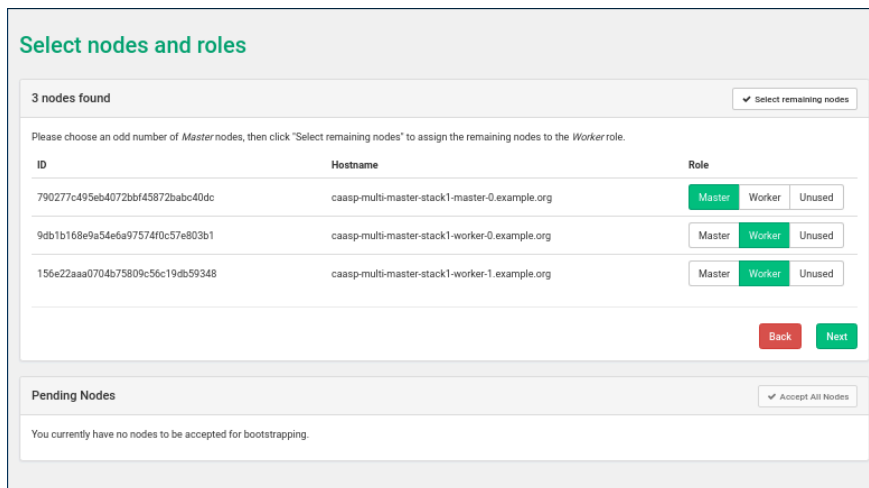
You don't need to boot each node by hand. More information on how to embed an AutoYaST profile in your PXE environment is available [here](#). Where `http://10.84.65.17/autoyast` is the real, generated path to the AutoYaST profile served by the dashboard.

Back Next

SUSE CaaS Platform Admin Node Install: Screen 3

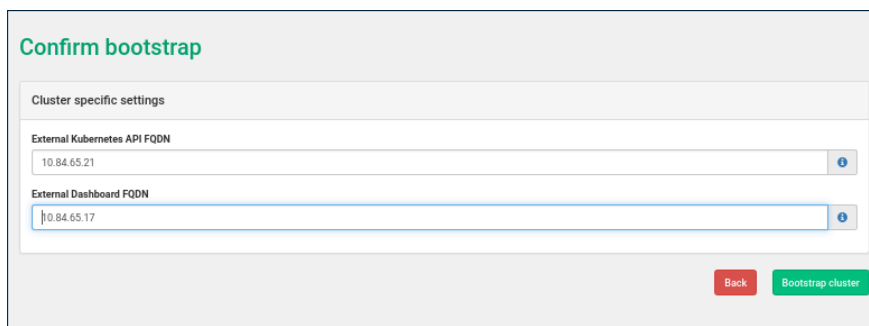


SUSE CaaS Platform Admin Node Install: Screen 4



SUSE CaaS Platform Admin Node Install: Screen 5

Set External Kubernetes API to LOADBALANCER_FLOATING_IP, External Dashboard FQDN to AD-MIN_NODE_FLOATING_IP



SUSE CaaS Platform Admin Node Install: Screen 6

Cluster Status

Summary

Total nodes 3 Updates Manual
Master nodes 1 # of nodes w/ outdated software 1
New nodes 0

Nodes

Status	ID	Hostname	Role	Actions
	790277c495eb4072bbf45872babc40dc	caasp-multi-master-stack1-master-0.example.org	<input checked="" type="checkbox"/> master	
	9db1b168e9a54e6a97574f0c57e803b1	caasp-multi-master-stack1-worker-0.example.org	<input type="checkbox"/> worker	Remove
	156e22aaa0704b75809c56c19db59348	caasp-multi-master-stack1-worker-1.example.org	<input type="checkbox"/> worker	Remove

Pending Nodes

You currently have no nodes to be accepted for bootstrapping.

SUSE CaaS Platform Admin Node Install: Screen 7

Cluster Status

Summary

Total nodes 3 Updates Manual
Master nodes 1 # of nodes w/ outdated software 1
New nodes 0

Nodes

Status	ID	Hostname	Role	Actions
	790277c495eb4072bbf45872babc40dc	caasp-multi-master-stack1-master-0.example.org	<input checked="" type="checkbox"/> master	
	9db1b168e9a54e6a97574f0c57e803b1	caasp-multi-master-stack1-worker-0.example.org	<input type="checkbox"/> worker	Remove
	156e22aaa0704b75809c56c19db59348	caasp-multi-master-stack1-worker-1.example.org	<input type="checkbox"/> worker	Remove

Pending Nodes

You currently have no nodes to be accepted for bootstrapping.

15.3 Enabling the Cloud Provider Integration (CPI) Feature

When deploying a CaaS Platform cluster using SUSE CaaS Platform OpenStack heat templates, the following CPI parameters can be set in `caasp-environment.yaml` or `caasp-multi-master-environment.yaml`.

`cpi_auth_url`

The URL of the keystone API used to authenticate the user. This value can be found on OpenStack Dashboard under *Access and Security > API Access > Credentials* (for example, `https://api.keystone.example.net:5000/`)

`cpi_domain_name`

Name of the domain the user belongs to.

`cpi_tenant_name`

Name of the project the user belongs to. This is the project in which SUSE CaaS Platform resources are created.

`cpi_region`

Name of the region to use when running a multi-region OpenStack cloud. The region is a general division of an OpenStack deployment.

`cpi_username`

Username of a valid user that has been set in keystone. Default: `admin`

`cpi_password`

Password of a valid user that has been set in keystone.

`cpi_monitor_max_retries`

neutron load balancer monitoring max retries. Default: `3`

`cpi_bs_version`

cinder Block Storage API version. Possible values are `v1`, `v2`, `v3` or `auto`. Default: `auto`

`cpi_ignore_volume_az`

Ignore cinder and nova availability zones. Default: `true`

`dns_nameserver`

Set this to the IP address of a DNS nameserver accessible by the SUSE CaaS Platform cluster.

Immediately after the SUSE CaaS Platform cluster comes online, and before bootstrapping, install the latest SUSE CaaS Platform 3.0 Maintenance Update using the following steps.

1. Register the SUSE CaaS Platform nodes for Maintenance Updates by following the instructions in [Section 15.4, "Register SUSE CaaS Platform Cluster for Software Updates"](#).
2. On each of the SUSE CaaS Platform nodes, install the latest Maintenance Update:

```
tux > sudo transactional-update
```

Verify that the Velum image packages were updated:

```
tux > sudo zypper se --detail velum-image  
i | sles12-velum-image | package      | 3.1.7-3.27.3 | x86_64 | update_caasp
```

Reboot the node:

```
tux > sudo reboot
```

3. Finally, when preparing to bootstrap using the SUSE CaaS Platform web interface, upload a valid trust certificate that can validate a certificate presented by keystone at the specified `keystone_auth_url` in the `System-wide certificate` section of Velum. If the SSL certificate provided by keystone cannot be verified, bootstrapping fails with the error `x509: certificate signed by unknown authority`.



Note

If your OpenStack endpoints operate on the Internet, or if the SSL certificates in use have been signed by a public authority, no action should be needed to enable secure communication with them.

If your OpenStack services operate in a private network using SSL certificates signed by an organizational certificate authority, provide that CA certificate as the system-wide certificate.

If your OpenStack service SSL infrastructure was self-signed during the installation of SUSE OpenStack Cloud Crowbar 9 (as is done by default), its CA certificate (with the file extension `.pem`) can be retrieved from the admin node in the `/etc/ssl/certs/` directory. The filename should match the node name of your primary controller node. Download this file and provide it as the system-wide certificate.

The CPI configuration settings match the values provided via the `caasp-environment.yaml` or `caasp-multi-master-environment.yaml` files. Verify that they are correct before proceeding.

Cloud provider integration ⓘ
Enable Disable

OpenStack Settings

Keystone API URL *

This is the URL of the keystone API used to authenticate the user. This value can be found on the OpenStack control plane under "Access and Security" > "API Access" > "Credentials".

Domain name *

Used to specify the name of the domain your user belongs to.

Domain ID

Used to specify the ID of the domain your user belongs to.

Project name *

Used to specify the name of the project where you want to create your resources.

Project ID

Used to specify the ID of the project where you want to create your resources.

Region name *

Used to specify the identifier of the region to use when running on a multi-region OpenStack cloud. A region is a general division of an OpenStack deployment.

Username *

Refers to the username of a valid user set in keystone.

Password *
 🔑
Refers to the password of a valid user set in keystone.

Subnet UUID for the SUSE® CaaS Platform private network *

Used to specify the identifier of the subnet you want to create your load balancer on. This value can be found on the OpenStack control panels, under "Network" > "Networks" - click on the respective network to get its subnets.

Floating network UUID

When specified will lead to the creation of a floating IP for the load balancer.

Load balancer monitor max retries

Cinder Block Storage API version

Ignore Cinder availability zone
 True False

15.4 Register SUSE CaaS Platform Cluster for Software Updates

Software updates are published for all registered users of SUSE CaaS Platform, and should always be enabled upon deploying a new cluster.

These steps may be performed on cluster nodes one at a time, or in parallel, making SSH connections as the `root` user with the password that was established in your `/usr/share/caasp-openstack-heat-templates/caasp-environment.yaml` file.



Note

If using a private SMT server for registration, use its hostname or IP address when running the commands below. Otherwise, use `scc.suse.com` to connect to SUSE's public registration server.

1. If this node was previously registered, deactivate its current registration:

```
tux > sudo SUSEConnect -d
tux > sudo SUSEConnect --cleanup
```

2. If you are registering with a private SMT server, install its SSL certificate or the related organizational CA in order to perform SMT operations securely.

```
tux > sudo curl SMT_SERVER/smt.crt \
-o /etc/pki/trust/anchors/registration-server.pem
tux > sudo update-ca-certificates
```

3. Establish the new system registration:

```
tux > sudo SUSEConnect --write-config --url https://SMT_SERVER \
-r REGISTRATION_CODE -e EMAIL_ADDRESS
```

The same registration code may be used for all the nodes in your cluster.

4. Test the registration and look for a status of Registered.

```
tux > sudo SUSEConnect --status-text
```

15.5 More Information about SUSE CaaS Platform

More information about the SUSE CaaS Platform is available at <https://documentation.suse.com/suse-caasp/3/single-html/caasp-deployment/> ↗

16 Installing SUSE CaaS Platform v4 using terraform

16.1 CaaSP v4 deployment on SOC using terraform.

More information about the SUSE CaaS Platform v4 is available at https://documentation.suse.com/suse-caasp/4.0/html/caasp-deployment/_deployment_instructions.html#_deployment_on_suse_openstack_cloud ↗



Note

For SOC deployments that support Octavia, set `export OS_USE_OCTAVIA=true` in the downloaded openstack rc file in order for the load balancing API requests to the octavia service instead of the networking service.

IV Setting Up Non-OpenStack Services

17 Deploying the Non-OpenStack Components 293

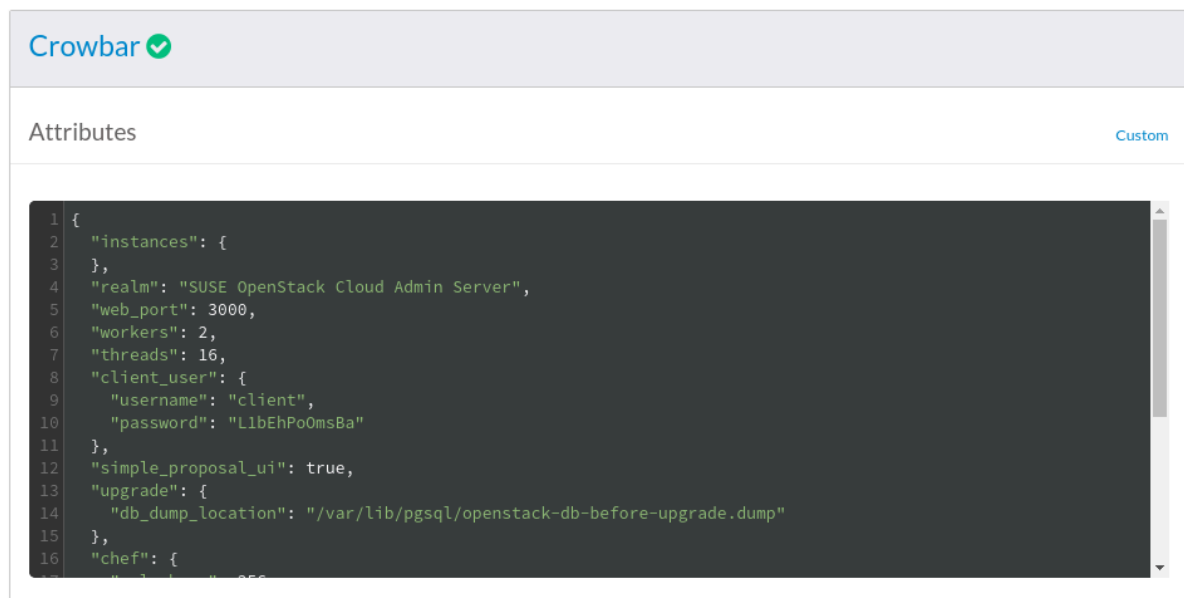
17 Deploying the Non-OpenStack Components

In addition to OpenStack barclamps, SUSE OpenStack Cloud includes several components that can be configured using the appropriate Crowbar barclamps.

17.1 Tuning the Crowbar Service

Crowbar is a self-referential barclamp used for enabling other barclamps. By creating a Crowbar proposal, you can modify the default number of threads and workers. This way, you can scale the admin server according to the actual usage or the number of available cores of the admin node.

● Edit Proposal



The screenshot shows the Crowbar web interface for editing a proposal. The title bar says "Crowbar" with a green checkmark. Below it, the word "Attributes" is displayed on the left, and "Custom" is on the right. The main content area is a dark-themed code editor showing a JSON configuration. The visible lines are:

```
1 {
2   "instances": {
3   },
4   "realm": "SUSE OpenStack Cloud Admin Server",
5   "web_port": 3000,
6   "workers": 2,
7   "threads": 16,
8   "client_user": {
9     "username": "client",
10    "password": "LibEhPo0msBa"
11  },
12  "simple_proposal_ui": true,
13  "upgrade": {
14    "db_dump_location": "/var/lib/pgsql/openstack-db-before-upgrade.dump"
15  },
16  "chef": {
```

FIGURE 17.1: THE CROWBAR BARCLAMP: RAW MODE

To change the default settings, create a Crowbar proposal and switch to the *Raw* view. Adjust then the `workers` and `threads` values. The number of threads should be set to the number of available cores. The default number of workers should be increased to 3 if the graphical interface becomes slow. Save and apply the changes using the appropriate buttons.

17.2 Configuring the NTP Service

The NTP service is responsible for keeping the clocks in your cloud servers in sync. Among other things, synchronized clocks ensure that the chef-client works properly. It also makes it easier to read logs from different nodes by correlating timestamps in them. The NTP component is deployed on the Administration Server automatically using the default settings. The NTP barclamp can be used to specify IP addresses of the external NTP servers and assign specific roles to the desired nodes. The following parameter can be configured using the NTP barclamp:

External servers

A comma-separated list of IP addresses of external NTP servers.

The NTP service consists of two different roles:

ntp-server

A node that acts as an NTP server for NTP clients in your cloud. There can be more than one node with the *ntp-server* role in your cloud. In this scenario, the NTP server nodes can communicate with each other and the specified external servers to keep their time in sync.

ntp-client

The *ntp-client* role can be assigned to any node. Nodes with the *ntp-client* role assigned to them keep their time in sync using NTP servers in your cloud.

17.3 Installing and using Salt

Crowbar can setup Salt on the admin node to be able to use `salt-ssh` from the admin node.



Note

Salt is not replacing Chef. It can be used in parallel to Chef to automate tasks on the nodes.



Note

Only `salt-ssh` can currently be used. Installing the Salt barclamp does not setup a full Salt stack with Salt Master and Salt Minions.

To be able to apply the Salt proposal, the <https://documentation.suse.com/sles/15-SP1/single-html/SLES-deployment/#sec-yast-install-addons> must be available on the node where the `salt-ssh` role will be applied (usually the admin node). After the Module is available, the barclamp can be applied.

From the node where the `salt-ssh` role is applied, Salt can be tested with:

```
# salt-ssh '*' test.ping
crowbar:
  True
storage1:
  True
controller:
  True
```

To list the available nodes visible to `salt-ssh`, do:

```
# salt-ssh -H
/etc/salt/roster:
-----
controller:
  192.168.192.81
crowbar:
  192.168.192.10
storage1:
  192.168.192.82
```

V Troubleshooting and Support

18 Troubleshooting and Support 297

18 Troubleshooting and Support

Find solutions for the most common pitfalls and technical details on how to create a support request for SUSE OpenStack Cloud Crowbar [here](#).

18.1 FAQ

If your problem is not mentioned here, checking the log files on either the Administration Server or the OpenStack nodes may help. A list of log files is available at *Book "Operations Guide Crowbar", Chapter 5 "Log Management", Section 5.4 "Log Files"*.

1 Admin Node Deployment

Q: *What to do if the initial SUSE OpenStack Cloud Crowbar installation on the Administration Server fails?*

A: Check the installation routine's log file at `/var/log/crowbar/install.log` for error messages.

Q: *What to do if the initial SUSE OpenStack Cloud Crowbar installation on the Administration Server fails while deploying the IPMI/BMC network?*

A: As of SUSE OpenStack Cloud Crowbar 8, it is assumed that each machine can be accessed directly via IPMI/BMC. However, this is not the case on certain blade hardware, where several nodes are accessed via a common adapter. Such a hardware setup causes an error on deploying the IPMI/BMC network. You need to disable the IPMI deployment running the following command:

```
/opt/dell/bin/json-edit -r -a "attributes.ipmi.bmc_enable" \  
-v "false" /opt/dell/chef/data_bags/crowbar/bc-template-ipmi.json
```

Re-run the SUSE OpenStack Cloud Crowbar installation after having disabled the IPMI deployment.

Q: *Why am I not able to reach the Administration Server from outside the admin network via the bastion network?*

A: If `route -n` shows no gateway for the bastion network, check the value of the following entries in `/etc/crowbar/network.json`: `"router_pref":` and `"router_pref":`. Make sure the value for the bastion network's `"router_pref":` is set to a *lower* value than `"router_pref":` for the admin network.

If the router preference is set correctly, `route -n` shows a gateway for the bastion network. In case the Administration Server is still not accessible via its admin network address (for example, `192.168.124.10`), you need to disable route verification (`rp_filter`). Do so by running the following command on the Administration Server:

```
echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
```

If this setting solves the problem, make it permanent by editing `/etc/sysctl.conf` and setting the value for `net.ipv4.conf.all.rp_filter` to `0`.

Q: *Can I change the host name of the Administration Server?*

A: No, after you have run the SUSE OpenStack Cloud Crowbar installation you cannot change the host name. Services like Crowbar, Chef, and the RabbitMQ will fail after changing the host name.

Q: *What to do when browsing the Chef Web UI gives a `Tampered with cookie error`?*

A: You probably have an old cookie in your browser from a previous Chef installation on the same IP address. Remove the cookie named `_chef_server_session_id` and try again.

Q: *How to make custom software repositories from an external server (for example a remote SMT or SUSE Manager server) available for the nodes?*

A: Custom repositories need to be added using the YaST Crowbar module:

1. Start the YaST Crowbar module and switch to the *Repositories* tab: `YaST > Miscellaneous > Crowbar > Repositories`.
2. Choose *Add Repositories*
3. Enter the following data:

Name

A unique name to identify the repository.

URL

Link or path to the repository.

Ask On Error

Access errors to a repository are silently ignored by default. To ensure that you get notified of these errors, set the Ask On Error flag.

Target Platform/Architecture

Currently only repositories for SUSE Linux Enterprise Server 12 SP4 on the x86-64 architecture are supported. Make sure to select both options.

4. Save your settings selecting *OK*.

2 OpenStack Node Deployment

Q: *How can I log in to a node as root ?*

A: By default you cannot directly log in to a node as root, because the nodes were set up without a root password. You can only log in via SSH from the Administration Server. You should be able to log in to a node with ssh root@NAME where NAME is the name (alias) of the node.

If name resolution does not work, go to the Crowbar Web interface and open the *Node Dashboard*. Click the name of the node and look for its *admin (eth0) IP Address*. Log in to that IP address via SSH as user root.

Q: *What to do if a node refuses to boot or boots into a previous installation?*

A: Make sure to change the boot order in the BIOS of the node, so that the first boot option is to boot from the network/boot using PXE.

Q: What to do if a node hangs during hardware discovery after the very first boot using PXE into the "SLEShammer" image?

A: The `root` login is enabled at a very early state in discovery mode, so chances are high that you can log in for debugging purposes as described in [Q:](#). If logging in as `root` does not work, you need to set the `root` password manually. This can either be done by setting the password via the Kernel command line as explained in [Q:](#), or by creating a hook as explained below:

1. Create a directory on the Administration Server named `/updates/discovering-pre`

```
mkdir /updates/discovering-pre
```

2. Create a hook script `setpw.hook` in the directory created in the previous step:

```
cat > /updates/discovering-pre/setpw.hook <<EOF
#!/bin/sh
echo "root:linux" | chpasswd
EOF
```

3. Make the script executable:

```
chmod a+x /updates/discovering-pre/setpw.hook
```

If you are still cannot log in, you very likely hit a bug in the discovery image. Report it at <http://bugzilla.suse.com/>.

Q: How to provide Kernel Parameters for the SLEShammer Discovery Image?

A: Kernel Parameters for the SLEShammer Discovery Image can be provided via the Provisioner barclamp. The following example shows how to set a `root` password:

1. Open a browser and point it to the Crowbar Web interface available on the Administration Server, for example <http://192.168.124.10/>. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it.
2. Open *Barclamps* > *Crowbar* and click *Edit* in the *Provisioner* row.
3. Click *Raw* in the *Attributes* section and add the Kernel parameter(s) to the `"discovery": { "append": "" }` line, for example;

```
"discovery": {
```

```
"append": "DISCOVERY_ROOT_PASSWORD=PASSWORD"
},
```

4. Apply the proposal without changing the assignments in the *Deployment* section.

Q: What to do when a deployed node fails to boot using PXE with the following error message: “Could not find kernel image: ../suse-12.2/install/boot/x86_64/loader/linux”?

A: The installation repository on the Administration Server at `/srv/tftpboot/suse-12.3/install` has not been set up correctly to contain the SUSE Linux Enterprise Server 12 SP4 installation media. Review the instructions at [Section 5.1, “Copying the Product Media Repositories”](#).

Q: Why does my deployed node hang at `Unpacking initramfs` during boot when using PXE?

A: The node probably does not have enough RAM. You need at least 4 GB RAM for the deployment process to work.

Q: What to do if a node is reported to be in the state `Problem`? What to do if a node hangs at “Executing AutoYast script: `/usr/sbin/crowbar_join --setup`” after the installation is finished?

A: Be patient—the AutoYaST script may take a while to finish. If it really hangs, log in to the node as `root` (see **Q:** for details). Check for error messages at the end of `/var/log/crowbar/crowbar_join/chef.log`. Fix the errors and restart the AutoYaST script by running the following command:

```
crowbar_join --start
```

If successful, the node will be listed in state `Ready`, when the script has finished.

In cases where the initial `--setup` wasn't able to complete successfully, you can rerun that once after the previous issue is solved.

If that does not help or if the log does not provide useful information, proceed as follows:

1. Log in to the Administration Server and run the following command:

```
crowbar crowbar transition $NODE
```

`NODE` needs to be replaced by the alias name you have given to the node when having installed it. Note that this name needs to be prefixed with `$`.

2. Log in to the node and run `chef-client`.

3. Check the output of the command for failures and error messages and try to fix the cause of these messages.
4. Reboot the node.

If the node is in a state where login from the Administration Server is not possible, you need to create a `root` password for it as described in [Direct root Login](#). Now re-install the node by going to the node on the Crowbar Web interface and clicking *Reinstall*. After having been re-installed, the node will hang again, but now you can log in and check the log files to find the cause.

Q: *Where to find more information when applying a barclamp proposal fails?*

A: Check the Chef client log files on the Administration Server located at `/var/log/crowbar/chef-client/d*.log`. Further information is available from the Chef client log files located on the node(s) affected by the proposal (`/var/log/chef/client.log`), and from the log files of the service that failed to be deployed. Additional information may be gained from the Crowbar Web UI log files on the Administration Server. For a list of log file locations refer to *Book "Operations Guide Crowbar", Chapter 5 "Log Management", Section 5.4 "Log Files"*.

Q: *How to Prevent the Administration Server from Installing the OpenStack Nodes (Disable PXE and DNS Services)?*

A: By default, the OpenStack nodes are installed by booting a discovery image from the Administration Server using PXE. They are allocated and then boot via PXE into an automatic installation (see [Section 11.2, "Node Installation"](#) for details). To install the OpenStack nodes manually or with a custom provisioning tool, you need to disable the PXE boot service and the DNS service on the Administration Server.

As a consequence you also need to provide an external DNS server. Such a server needs to comply with the following requirements:

- It needs to handle all domain-to-IP requests for SUSE OpenStack Cloud.
- It needs to handle all IP-to-domain requests for SUSE OpenStack Cloud.
- It needs to forward unknown requests to other DNS servers.

To disable the PXE and DNS services when setting up the Administration Server, proceed as follows:

PROCEDURE 18.1: **DISABLING PXE/DNS WHEN SETTING UP THE ADMINISTRATION SERVER**

The following steps need to be performed *before* starting the SUSE OpenStack Cloud Crowbar installation.

1. Create the file `/etc/crowbar/dns.json` with the following content:

```
{
  "attributes": {
    "dns": {
      "nameservers": [ "DNS_SERVER", "DNS_SERVER2" ],
      "auto_assign_server": false
    }
  }
}
```

Replace `DNS_SERVER` and `DNS_SERVER2` with the IP address(es) of the external DNS server(s). Specifying more than one server is optional.

2. Create the file `/etc/crowbar/provisioner.json` with the following content:

```
{
  "attributes": {
    "provisioner": {
      "enable_pxe": false
    }
  }
}
```

3. If these files are present when the SUSE OpenStack Cloud Crowbar installation is started, the Administration Server will be set up using external DNS services and no PXE boot server.

If you already have deployed SUSE OpenStack Cloud, proceed as follows to disable the DNS and PXE services on the Administration Server:

PROCEDURE 18.2: **DISABLING PXE/DNS ON AN ADMINISTRATION SERVER RUNNING CROWBAR**

1. Open a browser and point it to the Crowbar Web interface available on the Administration Server, for example `http://192.168.124.10/`. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it.

2. Open *Barclamps* > *Crowbar* and click *Edit* in the *Provisioner* row.
3. Click *Raw* in the *Attributes* section and change the value for *enable_pxe* to false:

```
"enable_pxe": false,
```

4. *Apply* the proposal without changing the assignments in the *Deployment* section.
5. Change to the *DNS* barclamp via *Barclamps* > *Crowbar* and click *Edit* in the *DNS* row.
6. Click *Raw* in the *Attributes* section. Change the value for *auto_assign_server* to false and add the address(es) for the external name server(s):

```
"auto_assign_server": false,  
"nameservers": [  
  "DNS_SERVER",  
  "DNS_SERVER2"  
],
```

Replace *DNS_SERVER* and *DNS_SERVER2* with the IP address(es) of the external DNS server(s). Specifying more than one server is optional.

7. *Save* your changes, but do not apply them, yet!
8. In the *Deployment* section of the barclamp remove all nodes from the *dns-server* role, but do not change the assignments for the *dns-client* role.
9. *Apply* the barclamp.
10. When the DNS barclamp has been successfully applied, log in to the Administration Server and stop the DNS service:

```
systemctl stop named
```

Now that the PXE and DNS services are disabled you can install SUSE Linux Enterprise Server 12 SP4 on the OpenStack nodes. When a node is ready, add it to the pool of nodes as described in [Section 11.3, "Converting Existing SUSE Linux Enterprise Server 12 SP4 Machines Into SUSE OpenStack Cloud Nodes"](#).

Q: *I have installed a new hard disk on a node that was already deployed. Why is it ignored by Crowbar?*

A: When adding a new hard disk to a node that has already been deployed, it can take up to 15 minutes before the new disk is detected.

Q: How to install additional packages (for example a driver) when nodes are deployed?

A: SUSE OpenStack Cloud Crowbar offers the possibility to install additional packages that are not part of the default scope of packages installed on the OpenStack nodes. This is for example required if your hardware is only supported by a third party driver. It is also useful if your setup requires to install additional tools that would otherwise need to be installed manually.

Prerequisite for using this feature is that the packages are available in a repository known on the Administration Server. Refer to [Q:](#) for details, if the packages you want to install are not part of the repositories already configured.

To add packages for installation on node deployment, proceed as follows:

1. Open a browser and point it to the Crowbar Web interface on the Administration Server, for example <http://192.168.124.10/>. Log in as user `crowbar`. The password is `crowbar` by default, if you have not changed it during the installation.
2. Go to *Barclamps > Crowbar* and click the *Edit* button for *Provisioner*.
3. Next click *Raw* in the *Attributes* page to open an editable view of the provisioner configuration.
4. Add the following JSON code *before* the last closing curly bracket (replace the `PACKAGE` placeholders with real package names):

```
"packages": {  
  "suse-12.2": ["PACKAGE_1", "PACKAGE_2"],  
}
```

Note that these packages will get installed on all OpenStack nodes. If the change to the Provisioner barclamp is made after nodes have already been deployed, the packages will be installed on the affected nodes with the next run of Chef or `crowbar-register`. Package names will be validated against the package naming guidelines to prevent script-injection.

3 Miscellaneous

Q: *How to change the keystone credentials after the keystone barclamp has been deployed?*

A: To change the credentials for the keystone administrator (admin) or the regular user (crowbar by default), proceed as follows:

1. Log in to the Control Node on which keystone is deployed as user root via the Administration Server.
2. In a shell, source the OpenStack RC file for the project that you want to upload an image to. For details, refer to [Set environment variables using the OpenStack RC file \(http://docs.openstack.org/user-guide/common/cli_set_environment_variables_using_openstack_rc.html\)](http://docs.openstack.org/user-guide/common/cli_set_environment_variables_using_openstack_rc.html) in the OpenStack documentation.
3. Enter the following command to change the PASSWORD for the administrator or the regular user (USER):

```
keystone-manage bootstrap --bootstrap-password PASSWORD \  
--bootstrap-username USER
```

For a complete list of command line options, run **keystone-manage bootstrap --help**. Make sure to start the command with a **Space** to make sure the password does not appear in the command history.

4. Access the keystone barclamp on the Crowbar Web interface by going to *Barclamps* > *OpenStack* and click *Edit* for the keystone barclamp.
5. Enter the new password for the same user you specified on the command line before.
6. Activate the change by clicking *Apply*. When the proposal has been re-applied, the password has changed and can be used.

Q: *How to add or change a configuration value for an OpenStack service?*

A: See [Chapter 14, Configuration Files for OpenStack Services](#).

18.2 Support

Before contacting support to help you with a problem on SUSE OpenStack Cloud, it is strongly recommended that you gather as much information about your system and the problem as possible. For this purpose, SUSE OpenStack Cloud Crowbar ships with a tool called **supportcon-**

fig. It gathers system information such as the current kernel version being used, the hardware, RPM database, partitions, and other items. **supportconfig** also collects the most important log files, making it easier for the supporters to identify and solve your problem.

It is recommended to always run **supportconfig** on the Administration Server and on the Control Node(s). If a Compute Node or a Storage Node is part of the problem, run **supportconfig** on the affected node as well. For details on how to run **supportconfig**, see <https://documentation.suse.com/sles/15-SP1/single-html/SLES-admin/#cha-adm-support>.

18.2.1 Applying PTFs (Program Temporary Fixes) Provided by the SUSE L3 Support

Under certain circumstances, the SUSE support may provide temporary fixes, the so-called PTFs, to customers with an L3 support contract. These PTFs are provided as RPM packages. To make them available on all nodes in SUSE OpenStack Cloud, proceed as follows.

1. Download the packages from the location provided by the SUSE L3 Support to a temporary location on the Administration Server.
2. Move the packages from the temporary download location to the following directories on the Administration Server:

“noarch” packages (*.noarch.rpm):

```
/srv/tftpboot/suse-12.4/x86_64/repos/PTF/rpm/noarch/  
/srv/tftpboot/suse-12.4/s390x/repos/PTF/rpm/noarch/
```

“x86_64” packages (*.x86_64.rpm)

```
/srv/tftpboot/suse-12.4/x86_64/repos/PTF/rpm/x86_64/
```

“s390x” packages (*.s390x.rpm)

```
/srv/tftpboot/suse-12.4/s390x/repos/PTF/rpm/s390x/
```

3. Create or update the repository metadata:

```
createrepo-cloud-ptf
```

4. The repositories are now set up and are available for all nodes in SUSE OpenStack Cloud except for the Administration Server. In case the PTF also contains packages to be installed on the Administration Server, make the repository available on the Administration Server as well:

```
zypper ar -f /srv/tftpboot/suse-12.4/x86_64/repos/PTF PTF
```

5. To deploy the updates, proceed as described in *Section 11.4.1, "Deploying Node Updates with the Updater Barclamp"*. Alternatively, run **zypper up** manually on each node.

A Using Cisco Nexus Switches with neutron

A.1 Requirements

The following requirements must be met to use Cisco Nexus switches with neutron:

- Cisco Nexus series 3000, 5000 or 7000
- All Compute Nodes must be equipped with at least two network cards.
- The switch needs to have the XML management interface enabled. SSH access to the management interface must be enabled (refer to the switch's documentation for details).
- Enable VLAN trunking for all neutron managed VLANs on the switch port to which the controller node running neutron is connected to.
- Before deploying neutron, check if VLAN configurations for neutron managed VLANs already exist on the switch (for example, from a previous SUSE OpenStack Cloud deployment). If yes, delete them via the switch's management interface prior to deploying neutron.
- When using the Cisco plugin, neutron reconfigures the VLAN trunk configuration on all ports used for the `nova-fixed` traffic (the traffic between the instances). This requires to configure separate network interfaces exclusively used by `nova-fixed`. This can be achieved by adjusting `/etc/crowbar/network.json` (refer to [Section 7.5, "Custom Network Configuration"](#)). The following example shows an appropriate configuration for dual mode, where `nova-fixed` has been mapped to conduit `intf1` and all other networks to other conduits. Configuration attributes not relevant in this context have been replaced with `...`

EXAMPLE A.1: EXCLUSIVELY MAPPING `NOVA-FIXED` TO CONDUIT `INTF1` IN DUAL MODE

```
{
  "attributes" : {
    "network" : {
      "conduit_map" : [
        ...
      ],
      "mode" : "single",
      "networks" : {
        "nova_fixed" : {
          ...
          "conduit" : "intf1"
        }
      }
    }
  }
}
```

```

    },
    "nova_floating" : {
        ...,
        "conduit" : "intf0"
    },
    "public" : {
        ...,
        "conduit" : "intf0"
    },
    "storage" : {
        ...,
        "conduit" : "intf0"
    },
    "os_sdn" : {
        ...,
        "conduit" : "intf0"
    },
    "admin" : {
        ...,
        "conduit" : "intf0"
    },
    "bmc" : {
        ...,
        "conduit" : "bmc"
    },
    "bmc_vlan" : {
        ...,
        "conduit" : "intf2"
    },
    },
    ...,
},
}
}

```

- Make a note of all switch ports to which the interfaces using the `nova-fixed` network on the Compute Nodes are connected. This information will be needed when deploying neutron.

A.2 Deploying neutron with the Cisco Plugin

1. Create a neutron barclamp proposal in the Crowbar Web interface.
2. As the *Plugin*, select `m12`.

3. As *Modular Layer 2 mechanism drive*, select cisco_nexus.
4. In *Modular Layer2 type drivers*, select vlan.
5. In the *Cisco Switch Credentials* table, enter the *IP Address*, the *SSH Port* number and the login credentials for the switch's management interface. If you have multiple switches, open a new row in the table by clicking *Add* and enter the data for another switch.

● Edit Proposal

Neutron !

Attributes Raw

DNS Domain

Networking Plugin

Plugin

Modular Layer 2 mechanism drivers

openvswitch

linuxbridge

cisco_nexus

vmware_dvs

cisco_apic_ml2

apic_gbp

Use Distributed Virtual Router setup

VLAN

Maximum number of VLANs

VXLAN

Start of VXLAN VNI range

End of VXLAN VNI range

Cisco Switch Credentials

IP Address	Port	Username	Password
Currently there are no switches assigned			
<input type="text" value="IP Address"/>	<input type="text" value="Port"/>	<input type="text" value="Username"/>	<input type="password" value="Password"/> 👁
<input type="button" value="Add"/>			

FIGURE A.1: THE NEUTRON BARCLAMP: CISCO PLUGIN

6. Choose whether to encrypt public communication (*HTTPS*) or not (*HTTP*). If choosing *HTTPS*, refer to *SSL Support: Protocol* for configuration details.
7. Choose a node for deployment and *Apply* the proposal.

8. Deploy nova (see [Section 12.11, “Deploying nova”](#)), horizon (see [Section 12.12, “Deploying horizon \(OpenStack Dashboard\)”](#)) and all other remaining barclamps.
9. When all barclamps have been deployed, return to the neutron barclamp by choosing *Barclamps > OpenStack > neutron > Edit*. The proposal now contains an additional table named *Assign Switch Ports*, listing all Compute Nodes.
For each Compute Node enter the switch it is connected to and the port number from the notes you took earlier. The values need to be entered like the following: 1/13 or Eth1/20.
10. When you have entered the data for all Compute Nodes, re-apply the proposal.



Important: Deploying Additional Compute Nodes

Whenever you deploy additional Compute Nodes to an active SUSE OpenStack Cloud deployment using the Cisco plugin with neutron, update the neutron barclamp proposal by entering their port data as described in the previous step.



Note: Verifying the Setup

To verify if neutron was correctly deployed, do the following:

1. Launch an instance (refer to the *User Guide*, chapter *Launch and manage instances* for instructions).
2. Find out which VLAN was assigned to the network by running the command **`openstack network show fixed`**. The result lists a *segmentation_id* matching the VLAN.
3. Log in to the switch's management interface and list the VLAN configuration. If the setup was deployed correctly, the port of the Compute Node the instance is running on, is in trunk mode for the matching VLAN.

B Documentation Updates

This chapter lists content changes for this document since the release of SUSE® OpenStack Cloud Crowbar 8.0.

This manual was updated on the following dates:

- *Section B.1, “April 2018 (Initial Release SUSE OpenStack Cloud Crowbar 8)”*

B.1 April 2018 (Initial Release SUSE OpenStack Cloud Crowbar 8)

Bugfixes

- In *Section 12.8, “Deploying glance”*, corrected the name of an example nova configuration file for custom settings (https://bugzilla.suse.com/show_bug.cgi?id=1077947 ↗).
- In *Section 12.8, “Deploying glance”*, updated the entries of a drop-down box (https://bugzilla.suse.com/show_bug.cgi?id=1073333 ↗).
- Numerous small fixes and corrections throughout the document (http://bugzilla.suse.com/show_bug.cgi?id=1073508 ↗, https://bugzilla.suse.com/show_bug.cgi?id=1073516 ↗).

Glossary of Terminology and Product Names

Active/Active

A concept of how services are running on nodes in a High Availability cluster. In an active/active setup, both the main and redundant systems are managed concurrently. If a failure of services occurs, the redundant system is already online, and can take over until the main system is fixed and brought back online.

Active/Passive

A concept of how services are running on nodes in a High Availability cluster. In an active/passive setup, one or more services are running on an active cluster node, whereas the passive node stands by. If the active node fails then the services are transferred to the passive node.

Administration Server

Also called Crowbar Administration Node. Manages all other nodes. It assigns IP addresses to them, boots them using PXE, configures them, and provides them the necessary software for their roles. To provide these services, the Administration Server runs Crowbar, Chef, DHCP, TFTP, NTP, and other services.

AMI (Amazon Machine Image)

A virtual machine that can be created and customized by a user. AMIs can be identified by an ID prefixed with `ami -`.

Availability Zone

An OpenStack method of partitioning clouds. It enables you to arrange OpenStack Compute hosts into logical groups. The groups typically have physical isolation and redundancy from other availability zones, for example, by using separate power supply or network equipment for each zone. When users provision resources, they can specify from which availability zone their instance should be created. This allows cloud consumers to ensure that their application resources are spread across disparate machines to achieve high availability if the hardware fails. Since the Grizzly release, availability zones are implemented via host aggregates.

AWS (Amazon Web Services)

A collection of remote computing services (including Amazon EC2, Amazon S3, and others) that together make up Amazon's cloud computing platform.

Barclamp

A set of Chef cookbooks, templates, and other logic. Used to apply a particular Chef role to individual nodes or a set of nodes.

ceilometer

Code name for *Telemetry*.

Cell

Cells provide a new way to scale Compute deployments. This includes the ability to have compute clusters (cells) in different geographic locations all under the same Compute API. This allows for a single API server being used to control access to multiple cloud installations. Cells provide logical partitioning of Compute resources in a child/parent relationship.

Ceph

A massively scalable, open source, distributed storage system. It consists of an object store, a block store, and a POSIX-compliant distributed file system.

Chef

An automated configuration management platform for deployment of your entire cloud infrastructure. The Chef server manages many of the software packages and allows the easy changing of nodes.

cinder

Code name for *OpenStack Block Storage*.

cloud-init

A package commonly installed in virtual machine images. It uses the SSH public key to initialize an instance after boot.

Cluster

A set of connected computers that work together. In many respects (and from the outside) they can be viewed as a single system. Clusters can be further categorized depending on their purpose, for example: High Availability clusters, high-performance clusters, or load-balancing clusters.

Cluster Partition

Whenever communication fails between one or more nodes and the rest of the cluster, a cluster partition occurs: The nodes of a cluster are split into partitions but still active. They can only communicate with nodes in the same partition and are unaware of the separated

nodes. As the loss of the nodes on the other partition cannot be confirmed, a *Split Brain* scenario develops.

Cluster Resource Manager

The main management entity in a High Availability cluster responsible for coordinating all non-local interactions. The *SUSE Linux Enterprise High Availability Extension* uses Pacemaker as CRM. Each node of the cluster has its own CRM instance. The instance running on the *Designated Coordinator (DC)* is the one elected to relay decisions to the other non-local CRMs and to process their input.

Compute Node

Node within a SUSE OpenStack Cloud. A physical server running a Hypervisor. A Compute Node is a host for guest virtual machines that are deployed in the cloud. It starts virtual machines on demand using `nova-compute`. To split virtual machine load across more than one server, a cloud should contain multiple Compute Nodes.

Container

A container is a storage compartment for data. It can be thought of as a directory, only that it cannot be nested.

Control Node

Node within a SUSE OpenStack Cloud. The Control Node is configured through the Administration Server and registers with the Administration Server for all required software. Hosts the OpenStack API endpoints and the OpenStack scheduler and runs the `nova` services—except for `nova-compute`, which is run on the Compute Nodes. The Control Node coordinates everything about cloud virtual machines: like a central communication center it receives all requests (for example, if a user wants to start or stop a virtual machine). It communicates with the Compute Nodes to coordinate fulfillment of the request. A cloud can contain multiple Control Nodes.

Cookbook

A collection of Chef recipes which deploy a software stack or functionality. The unit of distribution for Chef.

Corosync

The messaging/infrastructure layer used in a High Availability cluster that is set up with SUSE Linux Enterprise High Availability Extension. For example, the cluster communication channels are defined in `/etc/corosync/corosync.conf`.

Crowbar

Bare-metal installer and an extension of Chef server. The primary function of Crowbar is to get new hardware into a state where it can be managed by Chef. That means: Setting up BIOS and RAID, network, installing a basic operating system, and setting up services like DNS, NTP, and DHCP. The Crowbar server manages all nodes, supplying configuration of hardware and software.

Designated Coordinator (DC)

One *Cluster Resource Manager* in a High Availability cluster is elected as the Designated Coordinator (DC). The DC is the only entity in the cluster that can decide that a cluster-wide change needs to be performed. For example, fencing a node or moving resources around. After a membership change, the DC is elected from all nodes in the cluster.

DRBD (Distributed Replicated Block Device)

DRBD is a block device designed for building high availability clusters. The whole block device is mirrored via a dedicated network and is seen as a network RAID-1.

EBS (Amazon Elastic Block Store)

Block-level storage volumes for use with Amazon EC2 instances. Similar to OpenStack cinder.

EC2 (Amazon Elastic Compute Cloud)

A public cloud run by Amazon. It provides similar functionality to OpenStack Compute.

Ephemeral Disk

Ephemeral disks offer machine local disk storage linked to the life cycle of a virtual machine instance. When a virtual machine is terminated, all data on the ephemeral disk is lost. Ephemeral disks are not included in any snapshots.

Failover

Occurs when a resource fails on a cluster node (or the node itself fails) and the affected resources are started on another node.

Fencing

Describes the concept of preventing access to a shared resource by isolated or failing cluster members. Should a cluster node fail, it will be shut down or reset to prevent it from causing trouble. The resources running on the cluster node will be moved away to another node. This way, resources are locked out of a node whose status is uncertain.

Fixed IP Address

When an instance is launched, it is automatically assigned a fixed (private) IP address, which stays the same until the instance is explicitly terminated. Private IP addresses are used for communication between instances.

Flavor

The compute, memory, and storage capacity of `nova` computing instances (in terms of virtual CPUs, RAM, etc.). Flavors can be thought of as “templates” for the amount of cloud resources that are assigned to an instance.

Floating IP Address

An IP address that a Compute project can associate with a virtual machine. A pool of floating IP addresses is available in OpenStack Compute, as configured by the cloud operator. After a floating IP address has been assigned to an instance, the instance can be reached from outside the cloud by this public IP address. Floating IP addresses can be dynamically disassociated and associated with other instances.

glance

Code name for *OpenStack Image*.

Guest Operating System

An instance of an operating system installed on a virtual machine.

heat

Code name for *Orchestration*.

High Availability Cluster

High Availability clusters seek to minimize two things: system downtime and data loss. System downtime occurs when a user-facing service is unavailable beyond a specified maximum amount of time. System downtime and data loss (data is accidentally destroyed) can occur not only in case of a single failure. There are also cases of cascading failures, where a single failure deteriorates into a series of consequential failures.

horizon

Code name for *OpenStack Dashboard*.

Host

A physical computer.

Host Aggregate

An OpenStack method of grouping hosts via a common set of metadata. It enables you to tag groups of hosts with certain capabilities or characteristics. A characteristic could be related to physical location, allowing creation or further partitioning of availability zones. It could also be related to performance (for example, indicating the availability of SSD storage) or anything else that the cloud administrators deem appropriate. A host can be in more than one host aggregate.

Hybrid Cloud

One of several deployment models for a cloud infrastructure. A composition of both public and private clouds that remain unique entities, but are bound together by standardized technology for enabling data and application portability. Integrating SUSE Studio and SUSE Manager with SUSE OpenStack Cloud delivers a platform and tools with which to enable enterprise hybrid clouds.

Hypervisor

A piece of computer software, firmware or hardware that creates and runs virtual machines. It arbitrates and controls access of the virtual machines to the underlying hardware.

IaaS (Infrastructure-as-a-Service)

A service model of cloud computing where processing, storage, networks, and other fundamental computing resources are rented over the Internet. It allows the customer to deploy and run arbitrary software, including operating systems and applications. The customer has control over operating systems, storage, and deployed applications but does not control the underlying cloud infrastructure. Housing and maintaining it is in the responsibility of the service provider.

Image

A file that contains a complete Linux virtual machine.

In the SUSE OpenStack Cloud Crowbar context, images are virtual disk images that represent the contents and structure of a storage medium or device (such as a hard disk), in a single file. Images are used as a template from which a virtual machine can be started. For starting a virtual machine, SUSE OpenStack Cloud Crowbar always uses a copy of the image.

Images have both content and metadata; the latter are also called image properties.

Instance

A virtual machine that runs inside the cloud.

Instance Snapshot

A point-in-time copy of an instance. It preserves the disk state of a running instance and can be used to launch a new instance or to create a new image based upon the snapshot.

Keypair

OpenStack Compute injects SSH keypair credentials that are injected into images when they are launched.

keystone

Code name for *OpenStack Identity*.

libvirt

Virtualization API library. Used by OpenStack to interact with many of its supported hypervisors.

Linux Bridge

A software allowing multiple virtual machines to share a single physical NIC within OpenStack Compute. It behaves like a hub: You can connect multiple (physical or virtual) network interface devices to it. Any Ethernet frames that come in from one interface attached to the bridge is transmitted to all other devices.

Logical Volume (LV)

Acts as a virtual disk partition. After creating a *Volume Group (VG)*, logical volumes can be created in that volume group. Logical volumes can be used as raw block devices, swap devices, or for creating a (mountable) file system like disk partitions.

Migration

The process of moving a virtual machine instance from one Compute Node to another. This process can only be executed by cloud administrators.

Multicast

A technology used for a one-to-many communication within a network that can be used for cluster communication. Corosync supports both multicast and unicast.

Network

In the OpenStack Networking API: An isolated L2 network segment (similar to a VLAN). It forms the basis for describing the L2 network topology in a given OpenStack Networking deployment.

neutron

Code name for *OpenStack Networking*.

Node

A (physical) server that is managed by Crowbar.

nova

Code name for *OpenStack Compute*.

Object

Basic storage entity in OpenStack Object Storage, representing a file that you store there. When you upload data to OpenStack Object Storage, the data is neither compressed nor encrypted, it is stored as-is.

Open vBridge

A virtual networking device. It behaves like a virtual switch: network interface devices connect to its ports. The ports can be configured similar to a physical switch's port, including VLAN configurations.

OpenStack

A collection of open source software to build and manage public and private clouds. Its components are designed to work together to provide Infrastructure as a Service and massively scalable cloud computing software.

At the same time, OpenStack is also a community and a project.

OpenStack Block Storage

One of the core OpenStack components and services (code name: cinder). It provides persistent block level storage devices for use OpenStack compute instances. The block storage system manages the creation, attaching and detaching of the block devices to servers. Prior to the OpenStack Grizzly release, the service was part of nova-volume (block service).

OpenStack Compute

One of the core OpenStack components and services (code name: nova). It is a cloud computing fabric controller and as such, the main part of an IaaS system. It provides virtual machines on demand.

OpenStack Dashboard

One of the core OpenStack components or services (code name: horizon). It provides a modular Web interface for OpenStack services and allows end users and administrators to interact with each OpenStack service through the service's API.

OpenStack Identity

One of the core OpenStack components or services (code name: keystone). It provides authentication and authorization for all OpenStack services.

OpenStack Image

One of the core OpenStack components or services (code name: glance). It provides discovery, registration, and delivery services for virtual disk images.

OpenStack Networking

One of the core OpenStack components or services (code name: neutron). It provides “network connectivity as a service” between interface devices (for example, vNICs) managed by other OpenStack services (for example, Compute). Allows users to create their own networks and attach interfaces to them.

OpenStack Object Storage

One of the core OpenStack components or services (code name: swift). Allows to store and retrieve files while providing built-in redundancy and fail-over. Can be used for backing up and archiving data, streaming data to a user's Web browser, or developing new applications with data storage integration.

OpenStack Service

A collection of Linux services (or daemons) that work together to provide core functionality within the OpenStack project. This can be storing objects, providing virtual servers, or authentication and authorization. All services have code names, which are also used in configuration files, and command line programs.

Orchestration

A module (code name: heat) to orchestrate multiple composite cloud applications using file-based or Web-based templates. It contains both a user interface and an API and describes your cloud deployment in a declarative language. The module is an integrated project of OpenStack as of the Havana release.

PaaS (Platform-as-a-Service)

A service model of cloud computing where a computing platform and cloud-based application development tools are rented over the Internet. The customer controls software deployment and configuration settings, but not the underlying cloud infrastructure including network, servers, operating systems, or storage.

Pacemaker

An open source cluster resource manager used in SUSE Linux Enterprise High Availability Extension.

Port

In the OpenStack Networking API: An attachment port to an L2 OpenStack Networking network.

Private Cloud

One of several deployment models for a cloud infrastructure. The infrastructure is operated exclusively for a single organization and may exist on or off premises. The cloud is owned and managed by the organization itself, by a third party or a combination of both.

Private IP Address

See [Fixed IP Address](#).

Project

A concept in OpenStack Identity. Used to identify a group, an organization, or a project (or more generically, an individual customer environment in the cloud). Also called tenant. The term tenant is primarily used in the OpenStack command line tools.

Proposal

Special configuration for a barclamp. It includes barclamp-specific settings, and a list of nodes to which the proposal should be applied.

Public Cloud

One of several deployment models for a cloud infrastructure. The cloud infrastructure is designed for use by the general public and exists on the premises of the cloud provider. Services like applications, storage, and other resources are made available to the general public for free or are offered on a pay-per-use model. The infrastructure is owned and managed by a business, academic or government organization, or some combination of these.

Public IP Address

See [Floating IP Address](#).

qcow (QEMU Copy on Write)

A disk image format supported by the QEMU virtual machine manager. A qcow2 image helps to optimize disk space. It consumes disk space only when contents are written on it and grows as data is added.

qcow2 is a more recent version of the qcow format where a read-only base image is used, and all writes are stored to the qcow2 image.

Quorum

In a cluster, a *Cluster Partition* is defined to have quorum (is “quorate”) if it has the majority of nodes (or votes). Quorum distinguishes exactly one partition. It is part of the algorithm to prevent several disconnected partitions or nodes from proceeding and causing data and service corruption (*Split Brain*). Quorum is a prerequisite for *Fencing*, which then ensures that quorum is indeed unique.

Quota

Restriction of resources to prevent overconsumption within a cloud. In OpenStack, quotas are defined per project and contain multiple parameters, such as amount of RAM, number of instances, or number of floating IP addresses.

RC File (openrc.sh)

Environment file needed for the OpenStack command line tools. The RC file is project-specific and contains the credentials used by OpenStack Compute, Image, and Identity services.

Recipe

A group of Chef scripts and templates. Recipes are used by Chef to deploy a unit of functionality.

Region

An OpenStack method of aggregating clouds. Regions are a robust way to share some infrastructure between OpenStack compute installations, while allowing for a high degree of failure tolerance. Regions have a separate API endpoint per installation.

Resource

In a High Availability context: Any type of service or application that is known to the cluster resource manager. Examples include an IP address, a file system, or a database.

Resource Agent (RA)

A script acting as a proxy to manage a resource in a High Availability cluster. For example, it can start, stop or monitor a resource.

Role

In the Crowbar/Chef context: an instance of a *Proposal* that is active on a node.

In the *OpenStack Identity* context: concept of controlling the actions or set of operations that a user is allowed to perform. A role includes a set of rights and privileges. A user assuming that role inherits those rights and privileges.

S3 (Amazon Simple Storage Service)

An object storage by Amazon that can be used to store and retrieve data on the Web. Similar in function to OpenStack Object Storage. It can act as a back-end store for glance images.

SaaS (Software-as-a-Service)

A service model of cloud computing where applications are hosted by a service provider and made available to customers remotely as a Web-based service.

SBD (STONITH Block Device)

In an environment where all nodes of a High Availability cluster have access to shared storage, a small partition is used for disk-based fencing.

Security Group

Concept in OpenStack Networking. A security group is a container for security group rules. Security group rules allow to specify the type of traffic and direction (ingress/egress) that is allowed to pass through a port.

Sequence number (seqno)

A term from a MariaDB Galera Cluster which is used for replication. It's a 64-bit signed integer that the node uses to denote the position of a given transaction in the sequence.

Single Point of Failure (SPOF)

An individual piece of equipment or software which will cause system downtime or data loss if it fails. To eliminate single points of failure, High Availability systems seek to provide redundancy for crucial pieces of equipment or software.

SLEShammer

When you first boot a node in SUSE OpenStack Cloud Crowbar via PXE, it is booted with the SLEShammer image. This performs the initial hardware discovery, and registers the node with Crowbar. After you allocate the node, it is rebooted with a regular SLES installation image.

Snapshot

See *Volume Snapshot* or *Instance Snapshot*.

Split Brain

Also known as a “partitioned cluster” scenario. Either through a software or hardware failure, the cluster nodes are divided into two or more groups that do not know of each other. *STONITH* prevents a split brain situation from badly affecting the entire cluster.

Stateful Service

A service where subsequent requests to the service depend on the results of the first request.

Stateless Service

A service that provides a response after your request, and then requires no further attention.

STONITH

The acronym for “Shoot the other node in the head”. It refers to the fencing mechanism that shuts down a misbehaving node to prevent it from causing trouble in a cluster.

Storage Node

Node within a SUSE OpenStack Cloud. Acts as the controller for cloud-based storage. A cloud can contain multiple Storage Nodes.

Subnet

In the OpenStack Networking API: A block of IP addresses and other network configuration (for example, a default gateway, DNS servers) that can be associated with an OpenStack Networking network. Each subnet represents an IPv4 or IPv6 address block. Multiple subnets can be associated with a network, if necessary.

SUSE Linux Enterprise High Availability Extension

An integrated suite of open source clustering technologies that enables you to implement highly available physical and virtual Linux clusters.

SUSE OpenStack Cloud Administrator

User role in SUSE OpenStack Cloud Crowbar. Manages projects, users, images, flavors, and quotas within SUSE OpenStack Cloud Crowbar.

SUSE OpenStack Cloud Dashboard

The SUSE® OpenStack Cloud Crowbar Dashboard is a Web interface that enables cloud administrators and users to manage various OpenStack services. It is based on OpenStack Dashboard (also known under its codename horizon).

SUSE OpenStack Cloud Operator

User role in SUSE OpenStack Cloud Crowbar. Installs and deploys SUSE OpenStack Cloud Crowbar.

SUSE OpenStack Cloud User

User role in SUSE OpenStack Cloud Crowbar. End user who launches and manages instances, can create snapshots, and use volumes for persistent storage within SUSE OpenStack Cloud Crowbar.

swift

Code name for *OpenStack Object Storage*.

TAP Device

A virtual networking device. A TAP device, such as `vnet0` is how hypervisors such as KVM and Xen implement a virtual network interface card (vNIC). An Ethernet frame sent to a TAP device is received by the guest operating system. The tap option connects the network stack of the guest operating system to a TAP network device on the host.

Telemetry

A module (code name: `ceilometer`) for metering OpenStack-based clouds. The project aims to provide a unique point of contact across all OpenStack core components for acquiring metrics. The metrics can then be consumed by other components such as customer billing. The module is an integrated project of OpenStack as of the Havana release.

Tenant

See *Project*.

Unicast

A technology for sending messages to a single network destination. Corosync supports both multicast and unicast. In Corosync, unicast is implemented as UDP-unicast (UDPU).

User

In the OpenStack context, a digital representation of a person, system, or service who uses OpenStack cloud services. Users can be directly assigned to a particular project and behave as if they are contained in that project.

Veth Pair

A virtual networking device. The acronym veth stands for virtual Ethernet interface. A veth is a pair of virtual network interfaces correctly directly together. An Ethernet frame sent to

one end of a veth pair is received by the other end of a veth pair. OpenStack Networking uses veth pairs as virtual patch cables to make connections between virtual bridges.

VLAN

A physical method for network virtualization. VLANs allow to create virtual networks across a distributed network. Disparate hosts (on independent networks) appear as if they were part of the same broadcast domain.

VM (Virtual Machine)

An operating system instance that runs on top of a hypervisor. Multiple virtual machines can run on the same physical host at the same time.

vNIC

Virtual network interface card.

Volume

Detachable block storage device. Unlike a SAN, it can only be attached to one instance at a time.

Volume Group (VG)

A virtual disk consisting of aggregated physical volumes. Volume groups can be logically partitioned into logical volumes.

Volume Snapshot

A point-in-time copy of an OpenStack storage volume. Used to back up volumes.

vSwitch (Virtual Switch)

A software that runs on a host or node and provides the features and functions of a hardware-based network switch.

Zone

A logical grouping of Compute services and virtual machine hosts.